Inductive Modem Module (IMM)

User's Manual

Sea-Bird Electronics, Inc.
1808 136th Place NE
Bellevue, Washington 98005 USA
Telephone: 425/643-9866
Fax: 425/643-9954
E-mail: seabird@seabird.com

Manual Version 1.07, 2008-01-22

Limited Liability Statement

Extreme care should be exercised when using or servicing this equipment. It should be used or serviced only by personnel with knowledge of and training in the use and maintenance of oceanographic electronic equipment.

SEA-BIRD ELECTRONICS, INC. disclaims all product liability risks arising from the use or servicing of this system. SEA-BIRD ELECTRONICS, INC. has no way of controlling the use of this equipment or of choosing the personnel to operate it, and therefore cannot take steps to comply with laws pertaining to product liability, including laws which impose a duty to warn the user of any dangers involved in operating this equipment. Therefore, acceptance of this system by the customer shall be conclusively deemed to include a covenant by the customer to defend, indemnify, and hold SEA-BIRD ELECTRONICS, INC. harmless from all product liability claims arising from the use or servicing of this system.

Table of Contents

INDUCTIVE MODEM MODULE (IMM) 1

LIMITED LIABILITY STATEMENT 3

TABLE OF CONTENTS 4

1 IMM OVERVIEW AND CONCEPTUAL MODEL 7
1.1 SCOPE AND INTENT OF IMM CONCEPTUAL MODEL 7
2 IMM CONCEPTUAL MODEL 9
2.1 HARDWARE 9
2.1.1 Host 9
2.1.2 DPSK Transceiver with Tone Detect 9
2.1.3 Signal Detector 9
2.1.4 Inductive Cable Coupler 9
2.1.5 Power Input 10
2.1.6 HostFlag Digital Input 10
2.1.7 IMFlag Digital Output 10
2.2 NONVOLATILE STRUCTURES 10
2.2.1 Hardware Data 10
2.2.2 Configuration Data 10
2.2.3 Status Data 11
2.2.4 Event Counters 11
2.2.5 Host Data File 11
2.2.6 Sample Data 11
2.2.7 GData Reply Buffer 11
2.2.8 IM Reply Buffer 12
2.3 VOLATILE STRUCTURES 12
2.3.1 Operating Mode 12
2.3.2 Host Command Buffer 12
2.3.3 IM Command Buffer 12
3 OPERATING MODES 13
3.1 HOST SERVICE 13
3.2 IM SERVICE 13
3.3 IM RECEIVE 13
3.4 SLEEP 13
3.5 POWER CYCLING IMM 14
4 REPLY TAGS AND XML-LIKE FORMATTING 15
4.1 <EXECUTED/> TAG 15
4.2 ERROR TAGS 15
4.3 <CONFIMATIONREQUIRED/> TAG 16
4.4 <WARNING> TAGS 16
4.4.1 Warning Tag Reporting Potentially Serious Consequences 17
4.4.2 Warning Tag Reporting Unexpected Action by the IMM 17
4.5 <EXECUTING/> TAG 17
4.6 COMMENT TAGS 17

4.7 <HOSTSERVICE2MINTIMEOUT/> REPLY TAG 17
4.8 DATA REPLY TAGS 18
5 IMM-HOST INTERFACE IN HOST SERVICE MODE 19
5.1 ENTERING HOST SERVICE MODE 19
5.1.1 Entering Host Service Mode with Power Cycling 19
5.1.2 Entering Host Service Mode with Serial Line Transition 20
5.1.3 Entering Host Service Mode with HostFlag Transition 20
5.2 SENDING COMMANDS FROM HOST TO IMM 20
5.2.1 Terminating Commands to IMM 20
5.3 IMM PROMPTS AND ECHOING 20
5.4 BACKSPACE 20
5.5 TIMEOUT PARAMETERS IN HOST SERVICE MODE 20
5.5.1 TModem2 Maximum Wait for Start of IM Reply 21
5.5.2 TModem3 Maximum IM Receive Time 21
5.5.3 TModem4 21
5.6 EXITING HOST SERVICE MODE 21
6 SENDING COMMANDS TO REMOTE DEVICES 22
6.1 CAPTURING IM LINE 22
6.2 SENDING DATA 22
6.3 SENDING WAKE-UP TONE 22
6.4 RELEASING IM LINE 22
7 RECEIVING COMMANDS FROM REMOTE DEVICES 23
7.1 WAKING IMM FROM IM LINE 23
7.2 RECEIVING NON-HOST COMMANDS 23
7.2.1 Receiving Group and Global Non-Host Commands 23
7.3 DETECTING WAKE-UP TONES 23
7.4 RECEIVING HOST COMMANDS 23
7.4.1 Device ID and Serial Number Addressed Host Commands 24
7.4.2 Group Addressed and Global Host Commands 24
7.5 DISABLING SIGNAL DETECTOR 24
8 IMM-HOST INTERFACE IN IM SERVICE MODE 25
8.1 WAKING HOST WITH IMM 25
8.1.1 Waking Host with Hardware Handshaking (IMFlag) 25
8.1.2 Waking Host with a Serial Byte 26
8.1.3 Waking Host with a Serial Break Condition 26
8.2 CONFIRMING HOST WAKE-UP 26
8.2.1 Confirming Host Wake-up with HostPrompt 26
8.2.2 Confirming Host Wake-up with HostFlag Input 26
8.3 SENDING COMMAND TO HOST 26
8.4 TERMINATING COMMAND TO HOST 27
8.5 HOST REPLIES 27
8.5.1 Host Echo 27
8.5.2 Termination of Host Reply 27
8.5.3 Termination Character from Host 28
8.5.4 HostFlag Transition 28
8.5.5 Inter-character Delay Termination (THost4) 28
8.5.6 Host Return to Sleep 28
8.6 TIMEOUT PARAMETERS IN IM SERVICE MODE 28
8.6.1 THost0 Maximum Wait for Wake-up Confirmation 29
8.6.2 THost1 Delay after Confirm 29
8.6.3 THost2 Maximum Wait for Start of Reply 29

8.6.4 THost3 Maximum IM Reply Transmission Time 29
8.6.5 THost4 Inter-Character Delay Timeout 29
8.6.6 THost5 IMFlag Hold after Reply Termination 30
9 HANDLING OF BINARY DATA 31
10 USING SAMPLE DATA 32
10.1 ADDING AND APPENDING SAMPLES 32
10.2 RETRIEVING LIST OF SAMPLES 32
10.3 READING SAMPLES 33
10.4 ERASING SAMPLES 33
10.5 OVERWRITING SAMPLES AND MEMORY OVERFLOW 33
11 IMM COMMANDS 34
11.1 COMMAND ADDRESSING 34
11.2 SENDING COMMANDS 34
11.2.1 Interrupting Replies to IM Commands 35
11.3 GROUP COMMANDS 36
11.4 COMMAND SUMMARY TABLES 36
11.4.1 Config Type 1 Commands 36
11.4.2 Config Type 1 Blocked Commands 37
11.4.3 Config Type 2 Commands 37
11.5 DETAILED COMMAND DESCRIPTIONS 39
11.6 CONFIG TYPE 2 SET COMMANDS 55
12 INTERFACE MODE SHORTCUTS 62
13 ELECTRICAL SPECIFICATION 63
13.1 IMM CONNECTOR DESCRIPTION 63
13.2 HARDWARE I/O LINE CHARACTERISTICS 64
13.3 ABSOLUTE MAXIMUM RATINGS 64
13.4 POWER SUPPLY INPUT 64
13.5 POWER CONSUMPTION ESTIMATES 65
14 FIRMWARE REVISION HISTORY 66
15 SPECIFICATION REVISION HISTORY 68
16 INDEX 69

Tables

Table 4.1 Error Types 16
Table 5.1 Summary of Host Service Mode Interfacing 19
Table 8.1 Summary of IMM-Host Interface Options in IM Service Mode 25
Table 12.1 Command Addressing Types 34
Table 12.2 Commands Blocked from Group Addressing 36
Table 12.3 SIM Commands Described in SIM Documentation 36
Table 12.4 SIM Commands Not Implemented in Config Type 1 36
Table 12.5 IMM Commands Blocked in Config Type 1 37
Table 12.6 Config Type 2 Command Summary 37
Table 14.1 J1 Pinout RS-232 / Logic Level Serial 63
Table 14.2 J1 Pinout RS-485 63
Table 14.3 J2 Pinout (same for RS-232 and RS-485 versions) 63
Table 14.4 Minimum, Maximum, and Typical Resistance Values 64
Table 14.5 Minimum and Maximum Voltages 64
Table 14.6 Power Consumption Measurements (Firmware Version 0.98) 65

1 IMM Overview and Conceptual Model

The IMM is a low-power microprocessor peripheral that interfaces new or existing instruments capable of logic-level serial, RS-232, or RS-485[1] to inductive moorings. IMM telemetry is compatible with existing instruments and Surface Inductive Modems (SIMs) from Sea-Bird. The IMM includes advanced capabilities that allow instrument self-discovery, inductive line status reporting, cable coupler performance measurement, reduced restrictions on IM host devices, and a more convenient and robust communications interface. Most older Sea-Bird instruments require firmware or hardware upgrades to use these advanced features, but use of advanced features will not disturb older instruments on the same mooring.

The goals of an inductive modem system are:
- Serve as a minimally restrictive and highly reliable communications link.
- Provide specific features required to achieve scientific goals in spite of any limitations of the communications link.
- Provide simple and stable interfaces to help developers make reliable, maintainable, and reusable software at any scale.
- Provide advanced features to improve reliability and efficiency of system programming, setup, and maintenance.

1.1 Scope and Intent of IMM Conceptual Model

This section contains a high level view of the IMM conceptual model, and should be read by any developer using the IMM. The purpose of a conceptual model is to provide a view of the basic properties and structures of concepts. In an instrument, the conceptual model serves as a reference with which to describe what aspects of the instrument are affected by each possible operation, whether a user command accepted by a device or a firmware function executing in the device. Careful command and function descriptions using sound conceptual models help reduce unintended consequences, both by end users of instruments and by programmers. This can dramatically improve the ease of use and overall reliability of systems.

The IMM conceptual model must make sense in the context of models of individual instruments of all types from all manufacturers, as well as buoy systems and future network nodes. High-level users must design systems that communicate with and control a wide variety of devices. This task becomes more manageable if a high-level conceptual model may be applied to these devices.

An individual manufacturer cannot specify the system-level conceptual model. In fact, the IMM will be used by multiple systems with fundamentally different models. To accommodate system designers, the IMM model must be stable and sufficiently specified to allow designers of the high level systems to easily interface the IMM model with their systems. The IMM conceptual model serves as part of a bridge between the specific features and limitations of inductive modems and oceanographic instruments and the more generalized requirements of higher-level network models and interfaces. Laying the foundation of this bridge now helps prevent changes in the IMM's interface to host instruments. Addition of Host IDs, Host Data, and Sample Data storage to the IMM and potential for XML support in this data may allow host devices to conform to high-level interfaces without changing their firmware. This may prove valuable for system integrators who have limited influence over the firmware of the instruments generating data.

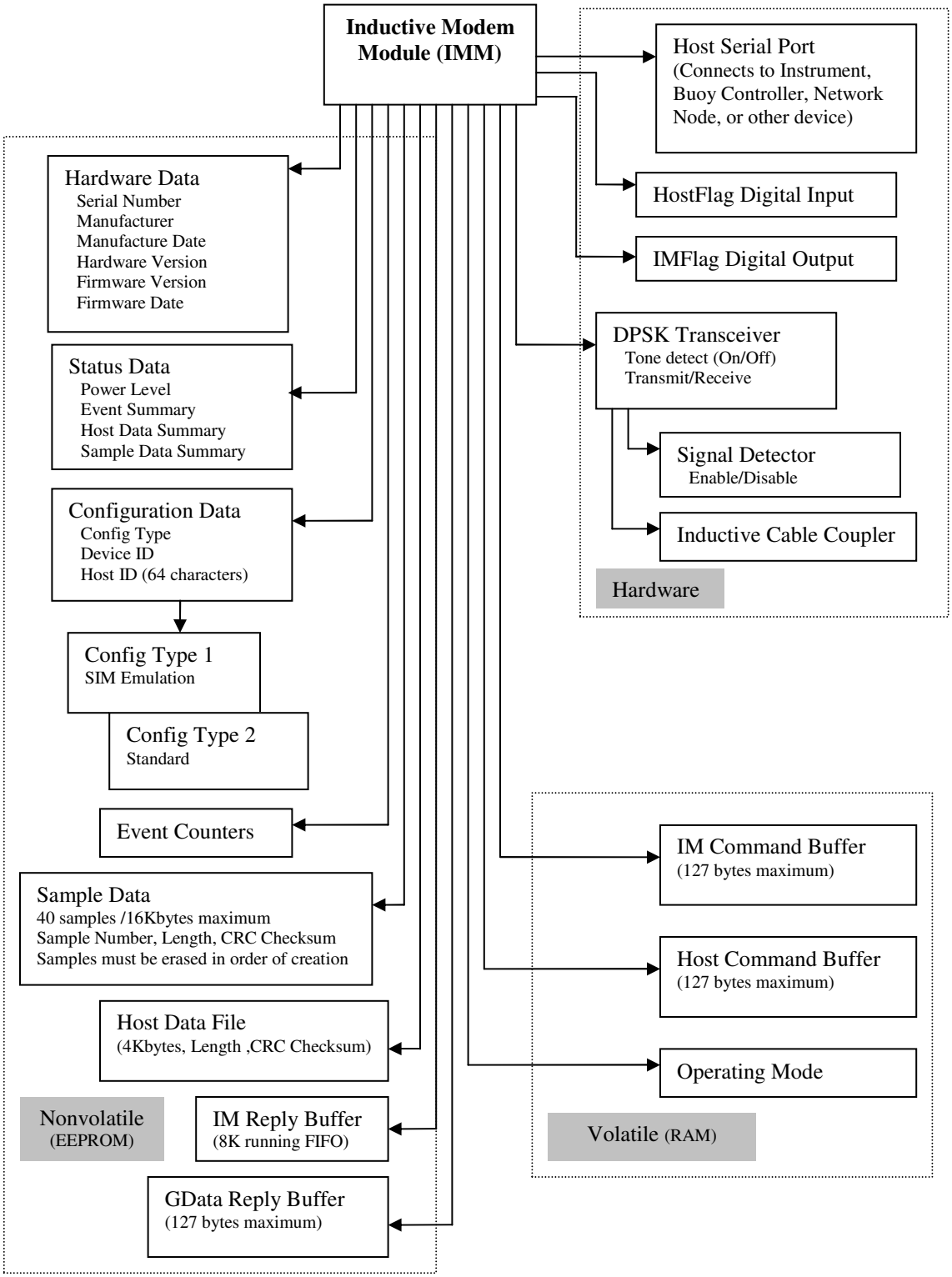
The IMM's conceptual model fundamentally differs from that of an instrument:
- It does not perform any environmental measurement or collect any scientific data directly.
- It is capable of controlling and communicating directly with an instrument or other device (some instruments may have this property).
- It is capable of communicating with a network of instruments through the IM line.

There is an attractive logical split between surface devices[2] and underwater[3] devices. The IMM model intentionally avoids making any conceptual split on this basis. This split may impose unnecessary limitations on the system that may interfere with future expansion.

---

[1] RS-232 and logic-level serial devices have a different part number from the RS-485 version. RS-485 devices may have longer lead times.
[2] Like the existing Surface Inductive Modem (SIM)
[3] Like the existing SBE 44 Underwater Inductive Modem (UIM)

Figure 1.1: Diagram of IMM Conceptual Model



Each item in the IMM conceptual model diagram is described in *Section 2 IMM Conceptual Model*.

2 IMM Conceptual Model

The IMM Model is divided into three logical sections: Hardware, Nonvolatile, and Volatile. The hardware section refers to physical devices and circuits. The Nonvolatile section contains structures stored in EEPROM that persist when the IMM loses power. The Volatile section consists of structures stored in RAM that are cleared or reset when the IMM loses power.

2.1 Hardware

2.1.1 Host

Every IMM has a single logical host. The host may be an instrument, a buoy controller, a node of a larger network, or some unknown hardware capable of serial communications. In Sea-Bird products like the SBE 39-IM or SBE 37-IM, the host instrument is the SBE 39 or SBE 37.

2.1.2 DPSK Transceiver with Tone Detect

The DPSK Transceiver is the electronic system that transmits and receives data with Differential Phase Shift Keying (DPSK) on the inductive mooring line. The transceiver is usually off; it turns on in response to a host command or an event from the Signal Detector.

2.1.3 Signal Detector

The Signal Detector is a low-power electronic system that detects the presence of a wake-up tone or DPSK transmission on the inductive mooring line. When the detector identifies a signal, it enables the DPSK Transceiver to receive and decode the signal. The Signal Detector can be disabled to save a small amount of power if it is not required[4].

2.1.4 Inductive Cable Coupler

The Inductive Cable Coupler is an electrical device, analogous to an antenna, which inductively couples the IMM's DPSK Transceiver to an inductive mooring cable. If the Inductive Cable Coupler is damaged or incorrectly installed, the communications system will not be reliable. The IMM has the ability to measure the performance of an assembled cable coupler.

There are two classes of Inductive Cable Couplers, 100-turn and 20-turn. The IMM is optimized for low power with the 100-turn coupler, but does work with the older, 20-turn coupler. Although different classes of cable couplers *may* be used on the same mooring, it is generally not recommended.

The Inductive Cable Coupler is also available in various forms, such as for integration with the instrument (as in the SBE 37-IM, 37-IMP, 39-IM, 16*plus*-IM, or 51), for use as an external coupler (as in the connection to a buoy), or for use as a test coupler for laboratory testing of the system.

Note: An IMM in the buoy can be operated without a cable coupler if using a direct cable connection configuration; this requires an external transformer in place of the cable coupler.

[4] The detector should be disabled only in an IMM with a host device that initiates all mooring communications. This is typical of many buoy controllers.

2.1.5 Power Input

The IMM may be directly connected to batteries, or power may be supplied by the host instrument. The IMM will not power up until the input voltage rises above 6.0V If the voltage applied to the IMM rises above 19V, a voltage limiting circuit activates. This circuit allows the IMM to operate with reduced efficiency at input voltages up to 30V. Systems with higher voltage power supplies should use the IMM Daughter board, which has a high-efficiency switching supply for 12V to 80V inputs.

The IMM has two internal, high-efficiency switching power supplies: one powers the low-voltage microcontroller, and the other provides a higher voltage for the DPSK transmitter. The transmitter power supply is bypassed and does not activate until the input voltage is greater than 9V, which is the maximum safe transmitting voltage. The IMM can measure the voltage of the transmitter power supply; this measurement nearly matches the input voltage when the input voltage is less than 9V.

2.1.6 HostFlag Digital Input

HostFlag is a digital input used to power on the IMM or terminate commands or replies to commands.

2.1.7 IMFlag Digital Output

IMFlag is an open collector digital output used to signal arrival of data or completion of an operation.

2.2 Nonvolatile Structures

2.2.1 Hardware Data

Hardware Data contains data describing the IMM, and can be modified only by the factory. A standard command (GetHD) allows read-only access to Hardware Data. Hardware data includes: Serial Number, Manufacturer, Hardware Manufacture Date, Hardware Version, Firmware Version, and Firmware Date.

Each IMM and IM-enabled instrument has a unique serial number greater than 100 and less than 2^{32}. Serial numbers are set by Sea-Bird and cannot be changed. In Sea-Bird IM-enabled instruments, the serial number of the inductive modem is the serial number of the instrument.

2.2.2 Configuration Data

Configuration data includes all settable parameters related to operation of the IMM, including all host communication settings. A standard command (GetCD) allows you to review Configuration Data. Configuration data does not include the IMM serial number, because the serial number cannot be changed, but the reply to GetCD includes the serial number for convenience.

Configuration data may be changed in the field, and may be updated by the host, by a PC with a special cable, or over the inductive modem. Some commands that edit configuration data must be confirmed by repeating the command before settings are changed.

The Configuration Type (Config Type) defines the set of available commands in the IMM. Configuration Types provide for backwards compatibility as well as future expansion. Config Type 1 emulates the now-obsolete SIM-coupled and SIM-direct modems. Config Type 2 is the standard IMM. Additional Configuration Types may be defined later for custom applications.

The Device ID is a two-digit number from 00 to 99 used to address the IMM. This number is user-programmable.

The Host ID is a user-programmable 64-character string provided to help describe and identify the host.

2.2.3 Status Data

Status Data, describing the IMM, contains data that change while deployed. Status Data is not always directly modifiable, but in general the ability to clear or reset elements of Status Data is provided. A standard command (GetSD) allows read access for the host or through the IM line. Multiple commands may affect Status Data; refer to the command summaries for specific details. Status data includes Power Level, Event Counters, Discovered by List, Host Data Summary, and Sample Data Summary.

2.2.4 Event Counters

Event Counters are an array of counters recording the number of occurrences of common timeouts, power-on resets, watchdog timer resets, and other conditions. Event Counters are analogous to the *Black Box* in airplanes; they help identify the root cause of a malfunction.

2.2.5 Host Data File

The Host Data File is a nonvolatile memory buffer, physically located on the IMM, reserved for storage of host-specific information. This buffer may be used to store anything, including binary data. Some users may store host setup information, an XML file describing the host, or JAVA scripts for processing data from the host. The buffer is limited to 4 Kbytes, and is read- and write-accessible from both the serial port and the inductive modem[5]. A Host Data summary is available with file length and checksum[6].

2.2.6 Sample Data

The Sample Data structure is a nonvolatile memory[7] space where the host may store data for later upload through the Inductive Modem. In buoy applications, if the host periodically writes to Sample Data, the buoy controller can upload the data from the IMM without disturbing the host. This data buffering allows host instruments to take measurements and report data independently of the buoy controller's update rate.

The Sample Data structure allows a variable maximum number of logical Samples (up to 40), each containing a variable number of bytes up to the full memory space (16KByte). Sample Data is read- and write-accessible from both the serial port and the inductive modem. A Sample Data summary is available with length and checksum[8] for each sample in memory.

A Sample ID is automatically assigned to each Sample as it is created. The Sample ID is a 32-bit unsigned integer, starting at 1 and incrementing each time a sample is created. There is no way to reset the assignment of Sample IDs in an IMM.

2.2.7 GData Reply Buffer

GData is a global IM command used to make time-synchronized measurements across multiple instruments of different types. It allows a single command to initiate a sample, even though different host instruments may require different command strings. The GData Reply Buffer is a 127 byte buffer that holds the reply to the last GData command. GData may also be sent as a group command or addressed to a specific instrument, e.g. !G0:GDATA or !00GDATA. Note that #00GDATA is generally an error - this causes the remote device with ID 00 to send the string "GDATA" to its host.

---

[5] True binary support is provided with the HostFileWrite command, which is accepted only from the serial port, not from the inductive modem. The HostFileAppend command is accessible from both the serial port and the inductive modem, but the Host-to-IMM termination sequence ("\r\n") may make storage of binary data with this command difficult.

[6] The Host Data File checksum is implemented in firmware V1.05 and later. The checksum will read as 0x00000000 for firmware versions prior to 1.05.

[7] Sample Data is implemented in EEPROM that has a nominal endurance of 1 million erase/write cycles. The Sample ID allows a rough estimate of the number of cycles the EEPROM had accumulated. If an instrument records 6 samples per hour, each of which uses the entire memory space of the EEPROM, manufacturer's data suggest 99.9 % of EEPROMs will last more than 20 years.

[8] Sample Data checksums are implemented in firmware V1.05 and later. Checksums will read as 0x00000000 for firmware versions prior to 1.05

2.2.8 IM Reply Buffer

The IMM has an 8K buffer for replies from the host to the Inductive Modem. Bytes are placed in the buffer as they are received from the host, and they are removed from the buffer as they are transmitted to the IM line. If the number of bytes in the buffer exceeds 8192, data is lost and a DPSK_BUF_OFLOW event is logged in the Event Counters. It is possible to transmit up to 20KBytes in a reply, but we recommend limiting replies to 8KBytes or less to avoid complications.

2.3 Volatile Structures

Volatile structures are reset when the IMM executes a power-on reset.

2.3.1 Operating Mode

IMM operating mode describes the action the IMM is performing at a particular moment; it is not a user-settable parameter. There are four operating modes: Host Service, IM Service, IM Receive, and Sleep. Power-off might be considered an additional operating mode in some applications. The IMM may be configured to enter Sleep mode or Host Service mode on power up.

2.3.2 Host Command Buffer

The Host Command Buffer is a 127Byte RAM buffer used only during Host Service mode. Characters received from the host are stored in this buffer until the termination sequence arrives (carriage return followed by line feed, as generated by the Enter key on Windows computers). When the termination sequence arrives, the IMM attempts to execute the contents of the Host Command Buffer as a command. All host commands must fit in this buffer or the IMM will generate an error. This size limitation does not apply to data sent after executing a command (e.g., the SampleAdd command, which prompts for up to 16KBytes of additional data).

2.3.3 IM Command Buffer

The IM Command Buffer is a 127Byte RAM buffer where incoming commands from the IM line are stored. All IM commands must fit in this buffer or the command will be rejected as transmission errors and will not be executed. This size limitation does not apply to replies to commands.

3 Operating Modes

Operating mode describes the action the IMM is performing at a particular moment; it is not a user-settable parameter. There are four operating modes: Host Service, IM Service, IM Receive, and Sleep. Power-off might be considered an additional operating mode in some applications.

3.1 Host Service

The IMM enters Host Service mode when communication is initiated by the IMM's host. In Host Service mode the host may send commands to the IMM to store data, change settings, or initiate communication with remote devices. The IMM remains in Host Service mode until it receives the PwrOff command from the host or the 2-minute timer expires.

During Host Service mode, the IMM cannot process commands arriving from the IM line. Instead, the IMM responds to properly addressed incoming IM signals with a <Busy/> tag, if the IMM is not processing a command from the host. When the IMM processes a command from the host in Host Service mode, the IM line is ignored. The IM receive buffer is cleared after processing each command from the host[9].

3.2 IM Service

The IMM enters IM Service mode when any valid command (including a command that relays data to the host) arrives from the IM line. The IMM remains in this mode until the command is executed. When in IM Service mode, the IMM does not respond to commands or wake-up attempts from the host, nor does it respond to additional commands arriving from the IM line. The IMM does not send <Busy/> replies to the IM line in IM Service mode.

3.3 IM Receive

The IMM enters IM Receive mode if the signal detector is enabled and a signal is detected on the IM line. A wake-up tone is not required to switch an IMM into IM Receive mode (Note: a wake-up tone is required for SBE 37, 16, and 44 to enter the analogous mode). Every 2 minutes the IMM tests the line for a valid signal[10]. If there is no signal, the IMM returns to Sleep mode. Arrival of a valid IM command resets the 2-minute timer. If the PwrOff command arrives from the IM line, the IMM pauses for 100 milliseconds, then switches to Sleep mode.

In IM Receive mode, the IMM responds to wake-up attempts from the host by switching directly to Host Service mode. At the end of Host Service mode, the IMM re-tests the IM line. If a valid signal is present, the IMM resets the 2-minute timer and resumes IM Service (Note: The same timer is used for IM Service and Host Service; it is reset on every mode change).

3.4 Sleep

Sleep mode is the IMM's low-power on-state. If the signal detector is enabled and a signal is identified on the IM line, the IMM switches to an intermediate mode for a few milliseconds to determine if the signal is a valid IM signal. If the signal is valid, the IMM switches to IM Receive mode; otherwise, it returns to Sleep.

[9] If the Host takes several seconds to enter data, as in a typical SampleAdd command, the IMM might act on the command stored in its IM receive buffer when it finishes processing the host command. This could cause the IMM to send a <Busy/> reply tens of seconds after the IM command actually arrived. This late reply might collide with other signals on the IM line. Clearing the IM receive buffer after each host command prevents this problem.

[10] This test should not interfere with incoming communications from the IM line.

3.5 Power Cycling IMM

The IMM contains EEPROM memory, which may be unreliable if power is removed during a write operation. The safest way to power down the IMM is to first use RELEASELINE to explicitly terminate the IM session (if the IM line was captured), then send the PWROFF command, wait 100 milliseconds, and then disconnect power and set the logic level serial input to floating or logic low state. This power-off procedure is primarily a concern for applications where the IMM will be power cycled in the field. If the IMM is power cycled in the field, remove power for at least 15 seconds before re-connecting. The IMM will not fully power on until the input voltage rises above 6 volts.

4 Reply Tags and XML-like Formatting

The IMM uses an XML-like format to pass information from the IMM to the host in Host Service mode and from the IMM to the remote device in IM Service mode. If the IMM connected to a host never operates in Host Service mode, the host will never see any XML-like tags. The XML-like format is not used when the IMM forwards incoming commands to a host device or when the IMM relays normal replies from a host device[11].

XML-like tags are designed to help host programmers identify common IMM responses and Errors with non-command-specific code. The purpose of the similarity to XML is to take advantage of a format already understood by most developers. IMM reply tags should not be considered XML.

4.1 <Executed/> Tag

The Executed tag indicates the IMM has completed processing the last command. It is always the last thing sent by the IMM after processing a command[12] (except the prompt), and the IMM never sends more than one Executed tag in response to a command[13]. If the IMM returns an Executed tag with no Error or ConfirmationRequired tags, the command was understood and successfully processed. The IMM does not add Executed tags to host responses to commands received from the IM line.

4.2 Error Tags

Error tags, of the form <Error type='Type' msg='Message'/>, provide a standard format for the IMM to report error conditions and common negative results. Each Error tag contains a type and a message (msg). The Error type is intended for machine parsing; the number of types will be kept to a minimum and tied to the IMM's conceptual model. The msg parameter is intended for system debugging. Users should avoid creating dependencies on the content of the msg parameter.

Any command processed in Host Service mode or received from the IM line in IM Service mode may result in one or more Error tags. The IMM may send more than one Error tag in response to a single command.

[11] The Error tag may be appended to host replies to incoming commands if there is an error in interacting with the host, such as a timeout condition.
[12] Config Type 1 does not use Executed tags for commands that mimic SIM behavior. In Config Type 2, the *ID= command does not use Executed tags.
[13] Note that a command sent to a remote device may get an Executed tag from the remote device within a <RemoteReply> tag - followed by an Executed tag from the IMM itself, which will be outside the RemoteReply tag.

Table 4.1 Error Types

Type	Description
INVALID COMMAND	• Command not recognized by instrument • Instrument command buffer overflowed (IMM 127 byte buffer)
INVALID ARGUMENT	• Improperly formatted argument • Argument out of fixed permissible range • Argument out of dynamic permissible range (IMM sample data - attempt to download data for sample ID not in memory)
NOT ALLOWED	• Valid command not allowed in current operating mode or configuration (e.g., not allowed while logging or not allowed when IM line not captured) • Valid command not allowed via specific interface (e.g., disabling IM receiver via IM interface)
FAILED	• Report of negative result (e.g., EEPROM test failed, CaptureLine command failed, cable coupler test failed)
ABORTED	• User aborted active process (e.g., escape key in IM transmit or discovery)
IM RECEIVE	• Inductive Modem received invalid characters
OVERFLOW	• Buffer or data structure overflowed and data was truncated
TIMEOUT	• Timing parameter violated
POWER FAIL	• Low battery or low power condition
FACTORY SETUP	• A factory setting was not entered; customers should never see this error

4.3 <ConfimationRequired/> Tag

The IMM sends the ConfirmationRequired tag in response to commands that must be repeated before they take effect. This is common when changing critical elements in Configuration Data, such as the serial type or baud rate. The ConfirmationRequired tag is often, but not always, accompanied by a Warning tag.

Example:

```
IMM>setbaudrate=19200
<WARNING>
IMM will power down
next power up after confirm will use
new baud rate
</WARNING>
<ConfirmationRequired/>
<Executed/>
IMM>setbaudrate=19200
<Executed/>
```

In the example, the first time SetBaudRate is sent, the ConfirmationRequired tag indicates the command did not take effect. When SetBaudRate is sent a second time, the ConfirmationRequired tag is not sent again, indicating the parameter is set.

4.4 <Warning> Tags

Warning tags have two significant functions. The first function is to report potentially serious consequences that may result from confirming and executing a command which returns the ConfirmationRequired tag. The second function is to notify the user or host program of unexpected actions by the IMM in response to a command. The function of a specific Warning tag depends on the command that generated the tag. Refer to the detailed command descriptions for interpretation of Warning tags for particular commands.

To ensure maximum upgradeability, any user program that parses IMM output should be capable of processing warning tags. Programs may depend on the existence of Warning tags in response to specific commands, but there must be no dependency on the content of Warning tags. Warning tags may expand to include a type parameter in the opening tag, similar to the type parameter in Error tags. Unlike Error tags, Warning tags are not self-closing.

4.4.1 Warning Tag Reporting Potentially Serious Consequences

The first function of warning tags is seen primarily in *Set* commands. Several Set commands return Warning tags when execution of the command may have serious consequences.

Example:

```
IMM>setserialtype=1
<WARNING>
IMM will power down
next power up after confirm will use
RS232 Serial interface
</WARNING>
<ConfirmationRequired/>
<Executed/>
IMM>setserialtype=1
<Executed/>
```

4.4.2 Warning Tag Reporting Unexpected Action by the IMM

This second function of Warning tags is used by the SampleAdd command. The SampleAdd command may return a warning tag if an old sample was automatically erased to make room for the new sample. This is not an error condition, since the new sample was created; but the host may need to identify this condition, so a Warning is generated. An example is provided in the detailed command description for the SampleAdd command.

4.5 <Executing/> Tag

The Executing tag indicates the IMM is processing the command, but additional time is required to complete the command. The Executing tag may be sent any number of times in response to a single command. In general, the IMM sends an Executing tag at least once per second if an operation takes more than 1 second to complete. The IMM does not interrupt open reply tags (e.g., <RemoteReply> or <SampleData>) to send Executing tags. The IMM does not send an Executing tag in response to # commands received from the IM line.

4.6 Comment Tags

Comment tags are presented in the standard XML format, as in <!--this is a comment-->. Comments carry data intended to help users become familiar with the operation of the IMM and keep track of the IMM state when controlling it with a terminal program. Comment tags are enabled when the DebugLevel is 3 or higher.

Comments are specifically not intended for machine parsing. To ensure maximum upgradeability, any user program that parses IMM output should be capable of ignoring comment tags and should have no dependency on the existence or content of comment tags.

4.7 <HostService2MinTimeout/> Reply Tag

If the IMM is in Host Service mode and the host does not enter a valid command to the IMM for 2 minutes, the IMM times out and goes back to sleep. The IMM sends the HostService2MinTimeout tag to indicate it has timed out. After sending the tag, but before the IMM goes to sleep, the IMM pauses for 250 milliseconds[14]. During this pause, the IMM will not respond to any input from the host. After the pause the IMM clears its serial buffers and goes to sleep immediately. Note that this reply tag is sent automatically by the IMM upon timeout, while all other reply tags are sent in response to a command.

The 2-minute timeout can be prevented by setting EnableHostFlagWakeup=1 and holding the HostFlag signal in a logic high state.

[14] The 250 mS pause guarantees that the IMM will not immediately wake up if the Host echoes the timeout tag.

4.8 Data Reply Tags

Data reply tags are returned in response to particular commands, and are closely linked to the IMM conceptual model and to Sea-Bird's evolving standard instrument model. Data reply tags for each command are described in detail with the command in *Section 11: IMM Commands*.

Examples of Data Reply Tags:

```
<EventSummary numEvents='[UINT16]'/>
<Sample ID='[HEX32]' Len='[HEX16]' CRC='[HEX32]'/>
<Power Vtx='[%.1f] '/ >
<LineStatus S='[STRING10]'/>
```

Reply Tag Value Definitions (note – brackets do not appear in data):

[UINTn] = unsigned n-bit integer (*Example*: UINT16 is 16-bit integer).
[HEXn] = unsigned n-bit integer in Hex format (*Example*: HEX16 is 16-bit integer in Hex).
[%.nf] = floating point number with n digits to right of decimal place (*Example: %*.1f is a number with 1 digit to the right of the decimal place, such as 8.5 or 19.5)
[STRINGn] = n-character string (*Example*: STRING10 is 10-character string)

5 IMM-Host Interface in Host Service Mode

Host Service mode allows the host to directly send configuration commands, store or retrieve data, or use the IMM to send information to other devices in the inductive loop. Host Service mode is initiated by the host[15]. After entering Host Service mode, the IMM remains ready to receive commands until the 2-minute timer expires or the host sends the PwrOff command.

Nearly all host-IMM interaction in Host Service mode takes the form of a command string sent by the host to the IMM, termination of the command, and the IMM's response to the command. The only exception is the <HostService2MinuteTimeout/> flag, which the IMM sends in response to a timeout condition. The host may send any number of commands to the IMM in a single Host Service session, provided the delay between completion of a command and the arrival of the next valid command is not longer than 2 minutes.

5.1 Entering Host Service Mode

There are three ways to enter Host Service mode. Each method can be independently disabled. Some applications require all three methods be disabled, preventing the IMM from ever entering Host Service mode[16].

The host may initiate Host Service mode at any time except when the IMM is in IM Service mode or during a blackout period after a PwrOff command, power cycle, or 2-minute timeout. When the IMM enters Host Service mode, it notifies the host with a prompt and the IMFlag signal. Any characters received after the IMM confirms wakeup are added to the command buffer, so the host should initialize its serial port before initiating Host Service mode. If spurious characters are sent during wakeup, the host may need to send the termination sequence ("\r\n") to clear the command buffer before sending the first command[17].

Table 5.1 Summary of Host Service Mode Interfacing

IMM **Wake-up**	**Confirm Wake-up**	**Command Termination**	**Reply Termination**	***IMM*** **Return To Sleep**
• Power-up • Serial byte or break • HostFlag high-low transition	• IMFlag pulled to GND • IMM prompt	• Termination character sequence (CR followed by LF, '\r\n') • HostFlag low-high transition • 2-minute timeout	• Termination character • Serial line break • Inter-character delay (2 seconds) • Prompt	• Power-down • PwrOff command • 2-minute timeout Note: IMM will not sleep if Hostflag is High and EnableHostFlagWakeup=1

5.1.1 Entering Host Service Mode with Power Cycling

This is intended for applications like simple buoy controllers, where all IM activity is initiated by the buoy controller. If EnableHostServeOnPwrUp=1, the IMM automatically enters Host Service mode when power is applied. Before turning off power, the host should send the PwrOff command to the IMM and wait 100 milliseconds. Power must remain off for at least 15 seconds to guarantee complete power down[18]. If the logic level serial input is used, the serial line must be either floating or in a logic low state or the IMM cannot properly power down. This is a good conservative approach to IM system design, because it guarantees a clean software state in the IMM each time the buoy controller initiates communication.

Enable/disable this feature with SetEnableHostServeOnPwrup=.

[15] Host Service mode can also be initiated by power cycling the IMM if EnableHostServeOnPwrup=1.
[16] Host Service mode is typically completely disabled in simple instruments that send data only in response to incoming commands from the IM line.
[17] The IMM ignores any strings from the host with fewer than 3 characters, excluding the termination sequence.
[18] Systems with high input voltages should wait longer than 15 seconds before re-applying power to the IMM. A system with 30V input should wait 1 minute.

5.1.2 Entering Host Service Mode with Serial Line Transition

If EnableSerialIMMWakeup=1, the IMM enters Host Service mode in response to a high-low transition on the serial input (logic level or RS232).

Enable/disable this feature with SetEnableSerialIMMWakeup=. Note that EnableSerialIMMWakeup cannot be set to 0 if EnableSignalDetector is set to 0, and vice versa. This prevents the IMM from entering a state from which it cannot wake up.

5.1.3 Entering Host Service Mode with HostFlag Transition

If EnableHostFlagWakeup=1, the IMM enters Host Service mode in response to a low-high transition on the HostFlag input. If EnableHostFlagWakeup=1, the IMM also wakes up in response to a static high condition on HostFlag when it prepares to go to sleep; this may occur when the IMM wakes in response to IM line signals, even if no commands address the IMM. When EnableHostFlagWakeup=1 and the IMM is in Host Service mode, the IMM will not time out after 2 minutes when HostFlag is high. Users may hardwire HostFlag high to guarantee the IMM will not exit Host Service mode.

Enable/disable this feature with SetEnableHostFlagWakeup=.

5.2 Sending Commands from Host to IMM

After waking the IMM and receiving wakeup confirmation (prompt and/or IMFlag pulldown), the host may send commands to the IMM. Bytes sent to the serial port are stored in the IMM command buffer until the command is terminated. If the IMM command buffer overflows, an error is generated upon termination of the command.

5.2.1 Terminating Commands to IMM

Commands may be terminated with the termination sequence or a low-high transition of the HostFlag signal. The termination sequence for commands from the host to the IMM is carriage return (CR, 0x0D or '\r') followed by line feed (LF, 0x0A or '\n'); the termination sequence cannot be disabled. HostFlag termination may be enabled/disabled with SetEnableHostFlagTerm=.

Note that the TermFromHost setting does not apply in Host Service mode; it applies only in IM Service when the host responds to data from the IMM.

5.3 IMM Prompts and Echoing

The IMM can send prompts and echo characters to the host. These features are important when using a terminal program to communicate with the IMM. The prompt may also serve as confirmation to the host that the IMM is awake and serving the host. Use SetEnablePrompt=1 to enable prompts; use SetEnableEcho=1 to enable echoing. Note that the prompt applies only to Host Service mode; the prompt is never sent to the host in IM Service mode.

5.4 Backspace

For convenience, the IMM can respond to the backspace character when a user sends commands with a terminal program. Use SetEnableBackspace=0 to disable the backspace action when transmitting binary data.

5.5 Timeout Parameters in Host Service Mode

The IMM has three timeout settings, in addition to the 2-minute timeout, that apply only in Host Service mode.

5.5.1 TModem2 Maximum Wait for Start of IM Reply

After transmitting an IM command to which a reply is expected, the IMM waits for the first byte of the reply until a timeout occurs[19]. This timeout and the setting to control the timeout period are called TModem2. Valid values for TModem2 are from 5 to 3000; units are tens of milliseconds (TModem2=100 means 1 second). Note: If the remote device is an IMM, the remote IMM initiates its reply before attempting to wake-up the remote host. TModem2 is analogous to THost2 on the IM receiver.

5.5.2 TModem3 Maximum IM Receive Time

After transmitting a command and receiving a start of reply indicator, the IMM listens to the reply until the reply terminates or a timeout occurs. This timeout and the setting to control the time period are called TModem3. TModem3 is analogous to THost3 on the IM receiver. Set TModem3 of the IMM transmitting the command to match or be slightly greater than THost3 of the IMM receiving the command. If the receiver's THost3 is much greater than the transmitter's TModem3, the transmitter might start to send a new command while the receiver is still replying to the first command. Valid values for TModem3 are from 100 to 60000; units are tens of milliseconds (TModem3=100 means 1 second). This allows IM receive time to range from 1 second to 600 seconds (ten minutes).

In Config Type 1, TModem3 has the same meaning, but different units. The units of TModem3 in Config Type 1 are seconds, with a maximum value of 600 seconds. If the GetCD response shows TModem3>600 , the actual delay in Config Type 1 is 600 seconds.

5.5.3 TModem4

TModem4 applies only to Config Type 1 *bii* binary commands (a special case for use exclusively with older SBE 44 and UIM devices). TModem4 is the maximum inter-character delay time before the IMM terminates reception of a reply to a bii command. This is analogous to the BINARYGAP setting on the SIM. Valid values for TModem4 are 0 to 3000; units are tens of milliseconds (TModem4=3000 means 30 seconds). The TModem4 setting exists in IMM firmware V1.02 and later. Note that inter-character delays cannot terminate IM reception in most communication, because the system may receive noise characters if the IMM is receiving while nothing is transmitting to the line.

5.6 Exiting Host Service Mode

The host should exit Host Service mode with the PwrOff command. The PwrOff command returns the IMM to Sleep mode. If the host removes the IMM's power supply, it should first send the PwrOff command and then wait 100 milliseconds before cutting power.

[19] If no IM carrier is detected, the IMM does not wait for THost2, but terminates immediately with a NO REPLY Error.

6 Sending Commands to Remote Devices

Commands can be sent to remote devices only when the IMM is in Host Service mode.

The Inductive Mooring line allows half-duplex communication, meaning only one device may transmit at a time. If two devices attempt to transmit at the same time, both messages will be corrupted. This limitation prevents the IMM from allowing a standard modem interface, with data transmitted and received simultaneously.

All communication on the Inductive Mooring line takes the form of commands and replies to commands. Only one command may be sent at a time. Not all commands generate replies. If a command requires a reply, the reply begins immediately after the command is received. No additional commands can be sent until the reply is completed. The maximum length of a command is 127 bytes. The maximum recommended size of a reply is 8 Kbytes. The absolute maximum duration of a reply is three minutes, which allows up to 20 Kbytes.

6.1 Capturing IM Line

When the IMM is in Host Service mode, the host may initiate communication on the IM line. Since the IM line is half-duplex (data flows in only one direction at a time), the host should first verify that no other device is using the IM line. The CaptureLine command checks for other devices transmitting on the IM line, and then reserves the IM line by transmitting a carrier signal. This process is called *capturing the line*[20].

The ForceCaptureLine command starts transmitting a carrier signal immediately, without first checking that no other device is using the IM line. This is appropriate for systems where only one device will attempt to capture the line.

6.2 Sending Data

After capturing the IM line, the host may send data to other devices on the line. The IMM supports the old !ii and #ii command formats, as well as new serial number and group command addressing schemes. See *Section 11: IMM Commands* for a complete description. Note that any outgoing IM command may return a NOT ALLOWED Error if the line is not captured, or a FAILED Error and a POWER FAIL Error if the transmit voltage is too low.

6.3 Sending Wake-up Tone

Older IM devices required a wake-up tone before they would power-up and start receiving data from the IM line. The IMM does not wait for a wake-up tone, but starts receiving as soon as a carrier signal is detected. To support legacy devices, the host may send a wake-up tone to the line with the SendWakeupTone command.

6.4 Releasing IM Line

After completing a session on the IM line, the host device may use the RELEASELINE command to close the IM session. RELEASELINE automatically sends PWROFF to the IM line to switch remote IM devices out of receive mode back into Sleep power mode. Alternatively, the host may send the PwrOff command to the IMM, which also causes the IMM to automatically send the power off signal to remote devices and release the IM line if it is captured, and then tells the IMM to return to Sleep mode.

Notes:
1. If the IM line is not properly released, all IM devices on the mooring stay awake for up to 2 minutes, causing an unnecessary drain on their batteries.

PwrOff tells the remote devices to return to Sleep mode, releases the line, and then tells the **IMM** to return to Sleep mode. **SendPwrOff** just tells the **remote devices** to return to Sleep mode. These are different commands.

[20] The probability that two devices will collide while capturing the line (i.e., both devices believe they have the line), given that they receive the CaptureLine command at the same time, is approximately 1 in 1000.

7 Receiving Commands from Remote Devices

The IMM receives data only if the Signal Detector is enabled. When another IM device captures the IM line, the IMM automatically enters IM Receive mode, unless the Signal Detector is disabled or the IMM is already in Host Service mode. If properly addressed data arrives when the IMM is in Host Service mode, the IMM responds with a <Busy/> signal. When a properly addressed command arrives from the IM line and the IMM is in IM Receive mode, the IMM enters IM Service mode and executes the command. If the command requires interaction with the IMM's host, as in # commands and certain global commands, the IMM enters IM Service mode and initiates communication with its host.

7.1 Waking IMM from IM line

The IMM has an internal Signal Detector with multiple internal wake-up states in which it detects incoming signals and differentiates between incoming IM transmissions and electrical noise caused by radar, lightening strikes, or other activity. When the IMM identifies an IM carrier on the line (i.e., another device has captured the line or sent a legacy wake-up tone), the IMM enters IM Receive mode.

7.2 Receiving Non-Host Commands

Commands addressing the IMM itself start with an !, except for certain global non-host commands. These non-host commands are executed without disturbing the host. When an IMM processes an incoming ! or non-host global command, it cannot respond to requests from the host.

7.2.1 Receiving Group and Global Non-Host Commands

Group commands and most non-host global commands do not allow a response. The *ID= and ID? commands are exceptions, but these commands may only be used when a single receiving device is coupled to the IM line. Note that any ! command can be made global by addressing it to group 0, as in !G0:Command. Group commands starting with ! are executed in the receiving IMM(s) in the same way as individually addressed commands, but replies to these commands are disabled.

7.3 Detecting Wake-Up Tones

The IMM does not require wake-up tones. The IMM supports wake-up tone detection for compatibility with some older UIM devices that sent a wake-up tone to prompt a SIM to start data upload.

The recommended alternative for new designs is to send a command from one IMM to the other to initiate a data upload, rather than sending a wake-up tone. A false wake-up tone event may be caused by electrical noise or radar signals, but a false incoming command is extremely unlikely.

Incoming wake-up tones are reported to the host only if EnableToneDetect=1. Reporting a wake-up tone has the same effect as a remote device sending '#iiWAKE-UP TONE DETECTED'. This means the IMM initiates communication with the host to report a wake-up tone in the manner in which it would report an incoming command.

Use SetEnableToneDetect= to enable/disable detection of wake-up tones.

7.4 Receiving Host Commands

When the IMM receives a host command (typically starting with #), the IMM wakes the host (see *Section 8: IMM-Host Interface in IM Service Mode*), and then the IMM forwards the command to the host. If a reply is required, the IMM then relays the host reply to the IM line. There are several optional methods of waking up the host, using the serial port or hardware handshake lines. There are also several timeout parameters governing communications with the host. The way in which the command is addressed generally determines whether or not a reply is required.

7.4.1 Device ID and Serial Number Addressed Host Commands

A command addressed with a device ID has the prefix #ii (for example, #01startsampling). A command addressed with a serial number has the prefix #Snnnnnnnn: (for example, #S70000009:startsampling). Commands addressed with these methods always have a reply. The reply may be empty if the host does not provide any data, or in the special case where THost2 is set to 0.[21] The process of waking the host, sending the incoming data, and relaying a reply is described in the next section.

7.4.2 Group Addressed and Global Host Commands

Group and Global host commands never allow an IM reply. The host may send a reply to its IMM, but the IMM will not transmit the data to the IM line. Some Group and Global commands cause the IMM to store the host reply in Sample Data or the GData Reply Buffer. Refer to the detailed command descriptions for these commands. Note that any # command can be made global by addressing it to group 0, as in #G0:Command. This works only if *Command* is understood by all host devices coupled to the IM line. Group commands starting with # are executed in the receiving IMM(s) in the same way as individually addressed commands, but replies to these commands are disabled.

7.5 Disabling Signal Detector

Entering IM Receive mode may be disabled with SetEnableSignalDetector=0. This prevents the IMM from detecting incoming signals (or noise) on the IM line, and is appropriate for applications where all IM communication initiates from a single device, as in most buoy controllers. Disabling the Signal Detector prevents detection of wake-up tones.

Note: The Signal Detector cannot be disabled if EnableSerialIMMWakeup=0, otherwise there would be a state from which the IMM would never wake up.

[21] Setting THost2 to 0 is a method of disabling host replies. See the detailed description for THost2 in *Section 8: IMM-Host Interface in IM Service Mode?*.

8 IMM-Host Interface in IM Service Mode

In IM Service mode the IMM will either respond to local command (e.g. !ii, !G0:) or initiate communications with the Host. The IMM responds to local commands without disturbing the Host. The IMM initiates communications with the host only in three situations. The first, and most common situation, is when a # command wih matching device ID arrives from the IM line, as would occur if the remote device that captured the IM line received a #ii command. The second situation is when a global command or matching group command arrives on the IM line, as would occur if the device that captured the line received SENDGDATA (a global command) or a #G0: group command. The third and least common situation is when EnableToneDetect=1 and a wake-up tone is detected, as would occur if the device that captured the line received the SENDWAKEUPTONE command.
In IM Service mode, the IMM-Host interface has seven stages: the IMM wakes the host, the host confirms wake-up, the IMM sends the command to the host, the IMM terminates the command, the host sends a reply to the IMM, the host terminates the reply, and then the host returns to sleep.

Table 8.1 Summary of IMM-Host Interface Options in IM Service Mode

Host **Wake-up**	**Confirm Wake-up**	**Command Termination**	**Reply Termination**	***Host*** **Return To Sleep**
• IMFlag pulled to GND • Serial byte ('\r') • Serial break	• Prompt from Host or break • HostFlag low-high transition • Maximum wait for confirm (THost0)	• Termination character (TermToHost) • Serial line break • Inter-character delay (50mS)[22]	• Termination character (TermFromHost) • HostFlag low-high transition • Inter-character delay (THost4)[23] • Maximum wait for start of reply (THost2) • Maximum IM reply transmission time (THost3)	• Automatic[24] • IMFlag release

8.1 Waking Host with IMM

In most applications, the host is asleep until a command arrives. The IMM uses one of the methods described below to wake up the host before sending the incoming command.

8.1.1 Waking Host with Hardware Handshaking (IMFlag)

Hardware handshaking allows the host to remain in a very low-power state while waiting for incoming commands on the DPSK line, relying on the open collector IMFlag signal to initiate its wakeup. In some applications it may be possible to control the host power supply directly with the IMFlag signal. More often, the IMFlag signal initiates the host power-up, and the host controls its own power-down sequence.

If EnableAutoIMFlag=1, the IMM automatically uses IMFlag to signal the host to wake-up. The IMFlag signal is pulled to ground when the host should wake-up. The IMM releases the IMFlag line after an adjustable delay (THost5, described below) when the data transfer process is completed.

Use SetEnableAutoIMFlag= to enable/disable all automatic switching of the IMFlag output. Disabling IMFlag applies to all operating modes.

[22] This delay would be measured by the host. The IMM sends the incoming data to the host in one burst with minimal separation between bytes.
[23] This is the inter-character delay specified by the Configuration Data setting THost4.
[24] The host may go back to sleep immediately after terminating its reply to the incoming command.

8.1.2 Waking Host with a Serial Byte

If EnableHostWakeupCR=1, the IMM sends a '/r' character (carriage return) to the host to generate a wake-up when initiating communication. If EnableHostWakeupBreak=1, the break condition is sent before the '/r' character.

Use SetEnableHostWakeupCR= to enable/disable the host wake-up carriage return feature.

8.1.3 Waking Host with a Serial Break Condition

If EnableHostWakeupBreak=1, the IMM sends a serial break condition to the host. The SerialBreakLen setting controls the duration of the break. If EnableHostWakeupCR=1, the break condition is sent before the '/r' character.

Use SetEnableHostWakeupBreak= to enable/disable the host wake-up break feature.

8.2 Confirming Host Wake-up

The IMM waits for confirmation that the host is awake before transmitting the command to the host. The THost0 setting provides an adjustable timeout for the confirmation process. If THost0 is set to 0, the host wake-up confirmation stage is disabled and the IMM sends the command to the host after the THost1 delay without waiting for confirmation.

8.2.1 Confirming Host Wake-up with HostPrompt

If EnableHostPromptConfirm=1, the host may confirm wake-up by sending a string matching the HostPrompt setting. HostPrompt is a string of 2 to 8 characters. Host prompt confirmation operates independently of other host wake-up options, except for the special case of THost0=0, which causes the IMM to ignore the host prompt settings.

Use SetEnableHostPromptConfirm= to enable/disable the host prompt confirm feature. Use SetHostPrompt= to set the host prompt.

If EnableHostPromptConfirm=1 and the IMM receives the wrong prompt from the host, the IMM logs the event as if it were a THost0 timeout.

8.2.2 Confirming Host Wake-up with HostFlag Input

If EnableHostFlagConfirm=1, the host may confirm wake-up with a low-high transition of the HostFlag input. HostFlag confirmation operates independently of other host wake-up options, except for the special case of THost0=0, which causes the IMM to ignore the HostFlag confirmation setting.

Use SetEnableHostFlagConfirm= to enable/disable the HostFlag confirm feature.

8.3 Sending Command to Host

The IMM allows an adjustable delay between receiving confirmation of wake-up from the host and transmission of the command to the host. This delay is set by THost1, described below. The THost1 delay always applies, even in the special case where host wake-up confirmation is disabled by setting THost0=0.

After the THost1 delay, the IMM sends the command to the host. There is no significant inter-character delay in transmission of data to the host.

8.4 Terminating Command to Host

After sending the command to the host, the IMM terminates the command. There are several options for terminating commands. Commands may be terminated with a single termination character, a carriage return followed immediately by a line feed (CRLF), or a serial line break[25]. The IMFlag signal cannot terminate commands to the host[26]. The host may interpret an inter-character delay greater than 50mS as termination.
Note that the IMM has separate settings for the termination sent to the host and the termination received from the host (see SetTermToHost= and SetTermFromHost=). TermToHost applies to termination of the command sent by the IMM to the host. Each setting is recorded in a single byte termination code. The IMM interprets any value less than 250 as a single byte termination character. The IMM interprets 253 as a serial line break (applicable for SetTermToHost only), 254 as CRLF, and 255 as no termination. If no termination is selected, the inter-character delay timeout should be shortened to act as a terminator. In CRLF termination, CR or LF may be sent to the device individually; only the CRLF combination acts as termination.

8.5 Host Replies

The host may begin its reply immediately after the IMM terminates the command. No inter-character delay is required in the host reply. The THost2 setting allows an adjustable timeout for the start of the host reply. If the host does not send at least one character (excluding echoed characters) in reply to the command within THost2, an error will be generated and communication cut off. There is an exception when THost2 is 0 - the IMM does not allow the host to reply at all; the IMM ignores the host response and behaves as if the host terminated its reply immediately.

The IMM starts transmitting the host reply to the IM line immediately. If bytes arrive from the host faster then they can be transmitted, they are stored in the Host Reply buffer. This is an 8 KByte FIFO buffer. No buffer overflow indication is provided for the host, but if the buffer overflows an error is generated and an event is logged in the Event Counters.

The IMM provides an inter-character-delay timeout (THost4) and a maximum reply time timeout (THost3). THost3 sets a hard limit on the maximum duration of a host reply at 180 seconds. This allows up to 20 Kbytes of data. Sea-Bird recommends limiting replies to 8 KBytes to avoid problems with the Host Reply buffer and the maximum reply time.

8.5.1 Host Echo

If the host echoes characters of the command sent by the IMM back to the IMM, the echoed command becomes part of the host reply. If EnableStripHostEcho=1, the IMM automatically deletes the host's echo of the command and any termination characters. Note that setting EnableStripHostEcho=1 will cause loss of 1 or more bytes of the host reply if the host does not echo the command.

Use SetEnableStripHostEcho= to enable/disable the strip host echo function.

Note: The IMM's echo must be disabled if the host echoes characters! Use SetEnableEcho=0 to disable the IMM's echo.

8.5.2 Termination of Host Reply

Host replies to commands may be terminated with a single termination character, a carriage return followed immediately by a line feed (CRLF), a low-high transition on the HostFlag input, or an inter-character delay timeout. Serial line breaks cannot terminate host replies. The IMM has separate settings for the termination sent to the host and the termination received from the host (see SetTermToHost= and SetTermFromHost=). TermFromHost applies to replies sent by the host to commands received from the IMM. If no termination is selected, the THost4 inter-character delay timeout should be shortened to act as a terminator. In CRLF termination, CR or LF may be sent to the device individually; only the CRLF combination acts as termination.

[25] Break condition termination is allowed only in the termination to the host, not from the host.
[26] Terminating commands to the host with IMFlag would cause problems for devices using IMFlag to control their power supplies.

The TermFromHost setting applies only in IM Service mode, when the host responds to data from the IMM. It does not apply in Host Service mode. Termination characters received from the host in IM Service mode are not included in the reply unless the termination character is CRLF (TermFromHost=254).

8.5.3 Termination Character from Host

The TermFromHost setting allows specification of a termination character or use of carriage return followed by line feed (CRLF) as a termination sequence. The setting is recorded in a single byte termination code. The IMM interprets any value less than 250 as a single byte termination character. The IMM interprets 254 as CRLF, and 255 as no termination. The remaining values are reserved. This feature operates independently of the other termination options, except for the special case where THost2=0, where the termination character setting is ignored.

Use SetTermFromHost= to configure the termination character for host replies. The command requires a decimal number as an argument.

8.5.4 HostFlag Transition

If EnableHostFlagTerm=1, the host may terminate replies to commands with a low-high transition of the HostFlag input. This feature operates independently of the other termination options, except for the special case where THost2=0, where the HostFlag termination setting is ignored.

Use SetEnableHostFlagTerm= to enable/disable HostFlag termination.

8.5.5 Inter-character Delay Termination (THost4)

Inter-character delay is the time between arrivals of individual characters in streaming serial data. The IMM allows an adjustable e inter-character delay timeout with the THost4 parameter. This timeout forces the end of the host reply. The THost4 timer activates after arrival of two characters. In instruments using hardware handshaking, termination characters, or break condition interfacing, the inter-character delay timeout acts as a safety mechanism, cutting off the host in case of malfunction or error[27]. If a host instrument cannot use handshaking, termination characters, or break conditions, THost4 may act as the primary reply termination mechanism. Inter-character delays are typically much shorter than maximum transmission lengths, so their use may save considerable line transmission time (and power) over a maximum reply time (THost3) timeout.

8.5.6 Host Return to Sleep

The host may go to sleep immediately after terminating its reply to the command. If the host requires a time delay between terminating its reply and release of the IMFlag output or arrival of the next command, the THost5 setting provides an adjustable delay.

8.6 Timeout Parameters in IM Service Mode

Inductive moorings support only one transmitting device at a time. A single device transmitting continuously might disable an entire mooring. Included in the safety features of the IMM are a variety of timeouts required to protect the mooring from damaged or malfunctioning host instruments.

In addition to the required timeouts, several optional timeouts and delays exist to allow the host to power-up, process commands, and power-down. Configuration of these timeouts and delays may be host-specific. The IMM is shipped with default timeout and delay values, but users should expect to configure these parameters to suit their application.

All timeouts and delays are part of the IMM's Configuration Data. Configuration Data can be updated by the host, with a PC and special cable or over the inductive modem. Configuration Data may be changed in the field.

[27] The inter-character delay timeout is not the primary safety mechanism, but it offers additional security through redundancy.

8.6.1 THost0 Maximum Wait for Wake-up Confirmation

Some hosts require time to initialize after wake-up. THost0 allows an adjustable timeout period for the IMM to wait for confirmation that the host is awake before the IMM assumes the host is awake. The delay does not apply to host-initiated communication. Valid values for THost0 are from 0 to 1000; units are tens of milliseconds (THost0=100 means 1 second).

If THost0 is set to a non-zero value, failure of the host to confirm wakeup before the timeout causes the IMM to log a THost0 event in the Event Counters. If the host interface uses the HostPrompt for wake-up confirmation, and the IMM receives the wrong prompt from the host, the IMM logs the event as if it were a THost0 timeout.

8.6.2 THost1 Delay after Confirm

THost1 provides an adjustable delay between the IMM receiving confirmation that the host is awake (or the THost0 timeout expiring) and the IMM sending data to the host. The delay does not apply to host-initiated communication. This delay is useful for slow instruments. THost1 can be set to 0 in most applications. Valid values for THost1 are from 0 to 1000; units are tens of milliseconds (THost1=100 means 1 second).

8.6.3 THost2 Maximum Wait for Start of Reply

After the IMM sends an incoming IM command to a host, it waits for a reply from the host. Some hosts may respond immediately, others may take seconds or longer to reply. THost2 is the maximum time the IMM waits for the first byte of reply from the host before assuming there is no reply. If the THost2 period passes with no reply bytes received from the host, the IMM responds to the IM line with an Error and logs a THost2 event in the Event Counters. Valid values for THost2 are from 100 to 3000, or 0; units are tens of milliseconds (THost2=100 means 1 second).

In the special case of THost2=0, the IMM assumes its host will never send a reply. All incoming # commands receive an automated empty reply after the host wakeup, wake-up confirmation, and transmission of the incoming command to the host is completed. No Error is reported or logged in this case. Setting THost2=0 is useful for simple unidirectional communication.

8.6.4 THost3 Maximum IM Reply Transmission Time

This is the maximum allowed duration of an IM reply to a # command, measured from the arrival of the command in the IMM to the termination of the IMM's transmission of the host's reply. This time limit acts on the 1200 baud reply on the IM line, as well as the higher speed host serial port. If the host fills the IMM's 8K reply buffer at 9600 baud, the IMM transmits from that buffer to the IM line at 1200 baud until either the buffer is empty or the THost3 timeout occurs. Any data remaining in the reply buffer after the THost3 timeout is discarded. When a THost3 timeout occurs, the host reply is cut off, an Error is forwarded to the IM line, and a THost3 event is logged in the Event Counters. Valid values for THost3 are from 100 to 18000; units are tens of milliseconds (THost3=100 means 1 second).

8.6.5 THost4 Inter-Character Delay Timeout

Inter-character delay is the time between arrivals of individual characters in streaming serial data. The inter-character delay timeout forces the end of the host's command or reply. This is useful with host devices that do not support termination characters in their replies. The inter-character delay timeout is enabled only after the arrival of the first character from the host. THost4 timeouts are reported as Errors only if the DebugLevel is set to 6 or higher. Valid values for THost4 are from 5 to 6000; units are tens of milliseconds (THost4=100 means 1 second).

8.6.6 THost5 IMFlag Hold after Reply Termination

Some Hosts require extra time to power down after replying to a command, especially if the IMFlag signal controls the host power supply. THost5 provides an adjustable delay between termination of the host's reply and release of the IMFlag by the IMM. The delay does not apply to host-initiated communication[28]. Valid values for THost5 are from 0 to 3000; units are tens of milliseconds (THost5=100 means 1 second).

[28] When the host initiates communication with the IMM, the IMFlag activates to signal IMM wake-up and deactivates to signal IMM entering Sleep mode or IM Receive mode, if there is an IM signal present at the end of Host Service mode.

9 Handling of Binary Data

The low-level IM transmission protocol reserves some characters to mark the start and end of transmissions, specify addressed instruments, and perform other IM-specific functions. These functions are abstracted from the user as much as possible to reduce system complexity. Unfortunately, the technique used prevents true binary data transmissions. When data sent by a host for transmission over the IM line contains characters reserved by the IM transmission protocol, these characters are automatically replaced by two characters, similar to the '\r' and similar codes used in many programming languages. This transformation increases the number of bytes that must be transmitted, thus increasing the expected transmission time.

A relatively simple rule allows specification of transmission times for binary data:

- The following binary values transmit as single characters -
  All values >=0x20
  TAB (0x09)
  LF (0x0A)
  CR (0x0D)
- All other binary values transmit as two characters and are automatically decoded when received[29]. The only observable effect for characters <0x20 is an increase in transmission time. There is no observable effect for TAB (0x09), LF (0x0A), CR (0x0D), and characters >=0x20.

It is the user's responsibility to prevent early termination of their commands from the host to the IMM (with the "\r\n" termination sequence) and in replies from the host to the IMM if termination characters are used. Termination characters less than 0x20 are allowed. Termination characters >= 0xFA (250 decimal) are NOT allowed.

[29] The backspace character (0x08) is interpreted as a command to delete the previously sent character if EnableBackspace=1. Use the SetEnableBackspace=0 command to disable the backspace character for binary data transfers.

10 Using Sample Data

The Sample Data structure is a nonvolatile memory space for data buffering between the host and a remote device connected to the inductive mooring. In buoy applications, data buffering allows host instruments to take measurements and report data independently of the update rate of the buoy controller. The IMM buffers data from the host instrument in the Sample Data structure between periodic uploads by the buoy controller (typically once per hour). The buoy controller may then use the Exclusive[30] command when uploading the larger data set. This reduces the total active time of the inductive mooring and extends battery life for other IM instruments on the mooring.

The Sample Data structure allows a variable maximum number of logical Samples (up to 40), each containing a variable number of bytes up to the full available memory space (16KByte). Sample Data is read- and write-accessible from both the serial port and the inductive modem. A Sample Data summary is available with length and checksum[31] for each sample in memory.

Responses to the GData command may be stored in the Sample Data structure. If EnableGDataToSample=1, the response to GData is diverted from the GData Reply buffer and is added as a new Sample in SampleData. This frees host devices from the 127 byte limit of the GData Reply Buffer.

A **Sample ID** is automatically assigned to each Sample as it is created. The Sample ID is a 32-bit unsigned integer, starting at 1 and incrementing each time a sample is created. There is no way to reset the assignment of Sample IDs in an IMM[32].

10.1 Adding and Appending Samples

There are three ways to add or append samples:

- The SampleAdd and SampleAppend commands allow addition of binary data. These commands are accepted only from the host, not from the IM interface.
- The SampleAddLine: and SampleAppendLine: command. These commands are accepted from the host or over the Inductive Modem, and allow addition of up to 100 bytes of data with standard termination. These commands are not intended for binary data. They do not include the CR and LF termination characters in the recorded data.
- Receiving a GDATA command from the IM line when EnableGDataToSample=1

10.2 Retrieving List of Samples

The SampleGetSummary and SampleGetList commands provide views of the contents of the Sample Data structure without downloading the contents of particular samples. SampleGetSummary returns a SampleDataSummary tag with the number of samples in memory, total length of data in memory, and available memory. SampleGetList returns a list identifying all samples in memory with the Sample ID, length and checksum for each sample.

Example:

```
IMM>samplegetsummary
<SampleDataSummary NumSamples='1' TotalLen='2529' FreeMem='13855'/>
<Executed/>
IMM>samplegetlist
<SampleList>
<Sample ID='0x00000008' Len='79' CRC='0x00000000'/>
<Sample ID='0x00000007' Len='2529' CRC='0x00000000'/>
</SampleList>
```

[30] The Exclusive command is not supported by the SBE 37, 16, 44, and 39-IM prior to firmware Version 2.0 and IMM prior to firmware Version 0.85.

[31] Sample Data checksums implemented in firmware V1.05 and later. The checksums will read as 0x00000000 if firmwre versions prior to 1.05

[32] Sample Data is implemented in EEPROM that has a nominal endurance of 1 million erase/write cycles. The Sample ID allows a rough estimate of the number of cycles the EEPROM had accumulated. If an instrument records 6 samples per hour, each of which uses the entire memory space of the EEPROM, manufacturer's data suggest 99.9 % of EEPROMs will last more than 20 years.

```
<Executed/>
```

10.3 Reading Samples

Samples are read with the SampleGetData: or SampleGetLast command.

- SampleGetData: allows data retrieval by sample ID, returning the data for a specific sample ID. The argument for SampleGetData: is a hexadecimal sample ID number.
- SampleGetLast retrieves data for the most recently added sample, which has the highest sample ID.

Data is output is enclosed in SampleData tags, including parameters for the sample ID and CRC code.

Example:

```
IMM>samplegetlast
<SampleData ID='0x7' LEN='16' CRC='0x0'>This is the data</SampleData>
IMM>samplegetdata:7
<SampleData ID='0x7' LEN='16' CRC='0x0'>This is the data</SampleData>
```

10.4 Erasing Samples

The Sample Data structure requires that samples be erased in the order in which they are created. This means the lowest Sample ID in memory must be erased first. There are three commands for erasing samples:

- SampleErase: only erases the oldest sample in memory, which has the lowest sample ID. The sample ID of the sample to erase must be specified; this prevents accidental deletion of samples.
- SampleEraseMultiple erases all samples starting at the specified ID and including all lower IDs.
- SampleEraseAll erases all samples in memory.

10.5 Overwriting Samples and Memory Overflow

If the Sample Data structure does not have enough memory to complete an add or append command, the data for that transaction is truncated. If an add command is received when the maximum number of samples already exist in memory, the oldest sample is automatically deleted. The maximum number of samples[33] may be set in Configuration Data with SetMaxNumSamples=xx, where xx is an integer from 1 to 40.

Example:

The Sample Data structure starts empty. A single sample is added that consumes the entire available memory space. Each time an Add command arrives, a new sample is created. All of these new samples are empty because there is no available memory. This continues until the maximum number of samples is reached. After the maximum is reached, the next Add command causes the original large sample to be erased. This immediately frees the memory that was taken by that original large sample, and the in-process add command successfully records data.

[33] If the maximum is set to a number smaller than the number of samples currently in memory, each SampleAdd or SampleAddN will delete only one old sample before creating a new sample, leaving the number of samples in memory greater than the specified maximum until a SampleErase:, SampleEraseMultiple:, or SampleEraseAll command removes some of the older samples.

11 IMM Commands

This section describes the IMM commands. Some commands are valid only when received from the host, others are valid from the host or the IM line. Many commands are valid only in a particular Configuration Type.

11.1 Command Addressing

Many IMM commands are executed by the IMM physically connected to the host sending commands. These are called Direct commands. It is important to distinguish between Direct commands and IM commands.

IM commands are transmitted over the IM line and executed in a remote device. IM command addressing is the process of identifying the device or devices on the inductive mooring that should execute a given command. The host addresses commands by prefixing them with either the ! or # character and the Device ID, serial number, or group number the command is intended for. The format for these prefixes is shown in the table of Command Addressing Types. Commands prefixed with an individual ID or individual serial number address a single instrument. Commands prefixed with a group number address a pre-defined group of instruments.

A small number of IMM commands cause transmission of global commands, which address all instruments on the mooring. Global commands do not use the # or ! prefixes.

Table 11.1 Command Addressing Types

Command Type	Prefix for Delivery to remote IMM	Prefix for Delivery to remote Host	Comments
Individual ID	!ii	#ii	ii = ID of remote IMM (ii = 0 – 99). Note: **ID must be two digits** (i.e., 00 is correct, 0 is incorrect)
Individual Serial Number	!Sx:	#Sx:	x = serial number of remote IMM ($100 < x < 2^{32}$)
Group	!Gn:	#Gn:	n = group number of remote IMMs (0 – 9, with 0 being the group of all IMMs). Notes: 1. ***G* prefix must be upper case for firmware versions prior to 1.06[34].** 2. Group commands do not allow replies.
Global (legacy)	Global commands do not use # or ! Global commands that have replies (such as ID? and *ID) are allowed only if there are exactly 2 IM devices on the mooring (Example: an IMM in the buoy and in 1 instrument). Global commands that have replies support older instruments and should be avoided when possible.		
Direct	Not applicable	Not applicable	Command sent by the IMM's host (e.g., CaptureLine)

11.2 Sending Commands

IM commands can be sent only when the IM line is captured. An attempt to send a command with a # or ! prefix when the line is not captured results in a NOT ALLOWED Error.

[34] Firmware versions prior to 1.06 listened for a reply to group commands if the G prefix was lower case. This firmware error was corrected in version 1.06.

11.2.1 Interrupting Replies to IM Commands

After sending an IM command, the host can interrupt the reply by sending the escape character (0x1B). This is useful if the host believes the remote device did not respond and the data in the RemoteReply is noise. Using the escape character stops the IMM from listening to the IM signal, but the remote device may still be transmitting. There is no way to interrupt an in-process transmission from a remote device. The host must not attempt to transmit to the IM line until the remote device stops transmitting. If more than one device transmits to the IM line at the same time, system behavior may be unpredictable.

11.3 Group Commands

Group commands allow addressing of pre-defined logical groups of instruments. For example, it may be useful to place all SBE 39-IMs in Group 1, all SBE 37-IMPs in Group 2, etc., allowing you to set parameters in all similar instruments at the same time. A maximum of nine groups may be defined (numbers 1 - 9). Group 0 is pre-defined as the group of all instruments. Individual IMMs cannot belong to more than two groups – Group 0 plus one other group.

Since group commands address multiple instruments, they do not allow replies. A command that ordinarily generates a reply does not generate a reply when sent as a group command.

Not all IMM commands are allowed with group commands. For example, all commands that start with *Get* are disabled, because these commands are not useful without a reply; some *Set* commands are also disabled. If a blocked group command is transmitted there is no error message, but the command is not executed in the remote instruments.

Table 11.2 Commands Blocked from Group Addressing

SetConfigType	TXTest	All commands that start with *Get*
SetInterfaceMode	EchoIM	All commands that start with *Test*
SetBaudRate	SetEnableSignalDetector	All commands that start with *HostFileGet*
SetDeviceID	SetEnableSerialIMMWakeup	All commands that start with *SampleGet*
SetSerialType	SetEnableHostFlagWakeup	

11.4 Command Summary Tables

11.4.1 Config Type 1 Commands

Config Type 1 emulated the SIM-coupled and SIM direct modems. Config Type 1 does not allow incoming commands from the IM line (signal detector is disabled). Several Config Type 1 commands either do not exist in Config Type 2, or behave differently from commands with the same name in
Config Type 2. These commands are designed to mimic the behavior of the older SIM board.

Table 11.3 SIM Commands Described in SIM Documentation and available in Config Type 1

PwrOn	HHMMSS=	GData
PwrOff	MMDDYY=	Dataii
StayOn	DDMMYY=	

Table 11.4 SIM Commands Not Implemented in Config Type 1

SIM Command	Comments
Baud=	Use SetBaudRate=.
DataNNMax=	No analogous setting for IMM.
RelayMax=	Use SetTHost3=, units are tens of milliseconds.
EchoOn	Use SetEnableEcho=1.
EchoOff	Use SetEnableEcho=0.
AutoPwrOn=	No analogous setting for IMM.
BinaryGap=	Use SetTHost4=, units are tens of milliseconds.

11.4.2 Config Type 1 Blocked Commands

All commands listed for Config Type 2 can be used in Config Type 1, except for blocked commands listed below. Blocked Commands generate a NOT ALLOWED error in firmware version 1.02 and higher.

Table 11.5 IMM Commands Blocked in Config Type 1

Captureline	SendWakeupTone	SetEnablePrompt	SetGDataStr
ForceCaptureLine	SetDeviceID	SetEnableSerialIMMWakeup	SetInterfaceMode
GetLineStatus	SetEnableHostFlagTerm	SetEnableSignalDetector	
Releaseline	SetEnableHostPromptConfirm	SetEnableHostFlagConfirm	

11.4.3 Config Type 2 Commands

Table 11.6 Config Type 2 Command Summary

Command	Allowed from Host	Allowed from IM	Allowed as Group	Comments
Outgoing IM commands or data (with !ii, #ii, !Sx:, #Sx:, !Gn:, or #Gn: prefix)	Captured Line Only	no	not applicable	
*ID=	Captured Line Only	no	no	Global IM command (legacy). Confirmation required. Returns Warning tag.
*Init	yes	no	no	Restores IMM to factory state. Confirmation required. Returns Warning tag.
CaptureLine	yes	no	no	
Data	no	yes	no	Similar to GetReply (legacy)
Disc	yes	no	no	Provisional Command – may change
EchoIM:	no	yes	no	
Exclusive:	no	yes	no	Use this command when transmitting from a SIM to an IMM. Use SENDEXCLUSIVE when transmitting from an IMM to other devices.
ForceCaptureLine	yes	no	no	
GData	no	yes	yes	Global IM command
GetCD	yes	yes	no	
GetConfigType	yes	yes	no	
GetEC	yes	yes	no	
GetHD	yes	yes	no	
GetHostFlag	yes	yes	no	
GetHostID	yes	yes	no	
GetLineStatus	yes	no	no	
GetReply	no	yes	no	Same as Data
GetSD	yes	yes	no	
HostFileAppendLine:	yes	yes	yes	
HostFileErase	yes	yes	no	Confirmation required.
HostFileGetData	yes	yes	no	
HostFileGetSummary	yes	yes	no	
HostFileWrite	yes	no	no	
ID?	Captured Line only	yes	no	Global IM command
MeasureLineSignal	yes	no	no	

Command	Allowed from Host	Allowed from IM	Allowed as Group	Comments
PwrOff	yes[35]	yes	yes	Global IM command to tell all IMMs to go to sleep (Note: This differs from SendPwrOff)
ReleaseLine	yes	no	no	
ResetEC	yes	yes	yes	
ResetEventCounters	yes	yes	yes	Same as ResetEC
SampleAdd	yes	no	no	
SampleAddLine:	yes	yes	yes	
SampleAppend	yes	no	no	
SampleAppendLine:	yes	yes	yes	
SampleErase:	yes	yes	no	
SampleEraseAll	yes	yes	yes	
SampleEraseMultiple:	yes	yes	no	
SampleGetData:	yes	yes	no	
SampleGetLast	yes	yes	no	
SampleGetList	yes	yes	no	
SampleGetSummary	yes	yes	no	
SendBreak	no	yes	yes	
SendExclusive:	yes	no	no	Use this command when transmitting from an IMM. Use EXCLUSIVE when transmitting from a SIM to an IMM.
SendGData	Captured Line only	no	no	Use this command when transmitting from an IMM to send the global GDATA command to the IM line. When using a SIM use GDATA instead.
SendPwrOff	Captured Line only	no	no	Tells remote devices to go to sleep, without powering off the IMM. (Note: This differs from PwrOff)
SendStayOn	Captured Line only	no	no	Use this command when transmitting from an IMM. Use STAYON when transmitting from a SIM
SendWakeupTone	Captured Line only	no	no	Use this command when transmitting from an IMM. Use PWRON when transmitting from a SIM
SetBaudRate=	yes	no	no	1200, 2400, 4800, 9600, 19200[36] Confirmation required. Returns Warning tag.
SetConfigType=	yes	no	no	1 or 2. Confirmation required. Returns Warning tag.
SetDebugLevel=	yes	yes	yes	0-9
SetDeviceID=	yes	yes	no	0-99, same as SetID=
SetEnableAutoIMFlag=	yes	yes	yes	0 or 1
SetEnableBackSpace=	yes	yes	yes	0 or 1
SetEnableBinaryData=	yes	yes	yes	0 or 1
SetEnableEcho=	yes	yes	yes	0 or 1
SetEnableFullPwrTX=	yes	yes	yes	0 or 1
SetEnableGDataToSample=	yes	yes	yes	0 or 1
SetEnableHostFlagConfirm=	yes	yes	no	0 or 1
SetEnableHostFlagTerm=	yes	yes	no	0 or 1
SetEnableHostFlagWakeup=	yes	yes	no	0 or 1

[35] In Host service, the PwrOff command automatically transmits PwrOff to the IM line if the IM line is captured.
[36] support for 38400 baud was removed in firmware version 1.05

Command	Allowed from Host	Allowed from IM	Allowed as Group	Comments
SetEnableHostPromptConfirm=	yes	yes	yes	0 or 1
SetEnableHostServeOnPwrUp=	yes	yes	yes	0 or 1
SetEnablePrompt=	yes	yes	yes	0 or 1
SetEnableHostWakeupCR=	yes	yes	yes	0 or 1
SetEnableHostWakeupBreak=	yes	yes	yes	0 or 1
SetEnableSerialIMMWakeup=	yes	yes	no	0 or 1. Confirmation required and returns Warning tag if set to 0.
SetEnableSignalDetector=	yes	no	no	0 or 1. Confirmation required and returns Warning tag if set to 0.
SetEnableStripHostEcho=	yes	yes	yes	0 or 1
SetEnableToneDetect=	yes	yes	yes	0 or 1
SetGDataStr=	yes	yes	yes	1-32 character string
SetGroupNumber=	yes	yes	yes	0-9
SetHostID=	yes	yes	yes	4-64 character string
SetHostPrompt=	yes	yes	yes	1-7 character string
SetID=	yes	yes	no	0-99, same as SetDeviceID=
SetIMFlag=	yes	yes	yes	0 or 1
SetInterfaceMode=	yes	no	no	1-14. Confirmation required and returns Warning tag. Note: For Versions <1.0, this command was SetConfigMode=.
SetMaxNumSamples=	yes	yes	yes	1-40
SetModemPrompt=	yes	yes	yes	1-7 character string
SetSerialBreakLen=	yes	yes	yes	1-255
SetSerialType=	yes	yes	no	0 – 2. Confirmation required and returns Warning tag.
SetTermFromHost=	yes	yes	yes	0-255
SetTermToHost=	yes	yes	yes	0-255
SetTHost0=	yes	yes	yes	0-1000
SetTHost1=	yes	yes	yes	0-300
SetTHost2=	yes	yes	yes	0, or 100-3000
SetTHost3=	yes	yes	yes	100-18000
SetTHost4=	yes	yes	yes	0-1000
SetTHost5=	yes	yes	yes	5-3000
SetTModem2=	yes	yes	yes	5-3000
SetTModem3=	yes	yes	yes	100-18000
SetTModem4=	yes	yes	yes	0-3000
StayOn	yes	yes	yes	Global IM command
Test20TurnCoupler	Captured Line only	yes	no	
TestCableCoupler	Captured Line only	yes	no	
TestEEPROM	yes	yes	no	
TXTest	no	yes	no	

11.5 Detailed Command Descriptions

Notes:

1. Command descriptions define whether a command is allowed from the IMM's host, from the Inductive Mooring (IM) line, and/or as a Group command.

2. When addressing a single IMM through the Inductive Mooring, the command formats shown are written for addressing the IMM by its ID. Any command that can be sent to a single IMM in this way can also be sent by addressing the IMM with its serial number. For example, the format shown for transmission of GetCD through the IM is !00getcd (transmit command to IMM with ID=00); it is also valid to transmit this command as !S70000019:getcd (transmit command to IMM with serial number = 70000019).

***ID=**	Sets Device ID of remote device(legacy). Use SetDeviceID= with serial number addressing in new designs.		
Allowed from / Format:	Host? Yes (Captured Line Only). *id=01	IM? No	Group? No
This is a global command that returns data. It must not be used if more than one instrument is coupled to the IM line. This command does not use reply tags.			
Modifies:	Device ID of all instruments on the mooring.		
Requires:	There must be exactly one receiving IM device on the mooring.		
Effects:	Sets the Device ID		
Returns:	WARNING tag		
Returns:	ConfirmationRequired tag		
Example:	IMM>*id=01 <RemoteReply><WARNING> Repeat Command to set id=01 Do not repeat if more than one IM device is coupled to the line!</WARNING> <ConfirmationRequired/> <Executed/></RemoteReply> <Executed/> IMM>*id=01 <RemoteReply><Executed/></RemoteReply> <Executed/>		

***Init**	Resets IMM connected to host to factory state		
Allowed from / Format:	Host? Yes *init	IM? No	Group? No
Modifies:	All IMM settable parameters - Serial Type, Baud Rate, Configuration Data, Host File, Sample Data, GData String, Event Counters, etc.		
Requires:	Nothing		
Effects:	Resets the IMM to the factory default state.		
Returns:	WARNING tag		
Returns:	ConfirmationRequired tag		
Example:	IMM>*init <WARNING> IMM will revert to factory state next power after confirm up will use RS232 Serial at 9600 baud Contents of Sample Data and HostFile will be erased Device ID will reset to 00 Event Counters will reset </WARNING> <ConfirmationRequired/> <Executed/> IMM>*init <Executed/>		

CaptureLine	Checks for other devices transmitting on IM line, and then reserves IM line by transmitting a carrier signal.		
Allowed from / Format:	Host? Yes captureline	IM? No	Group? No
Modifies:	DPSK Transceiver State		
Requires:	Nothing		

Effects:	Attempts to start transmitting a carrier on the IM line, first checking for other transmitting devices. If the line is captured when this command arrives, the IMM releases the line for 100mS and attempts to capture it again.
Returns:	FAILED error if another device is transmitting on the line, no error if the line is captured.
Returns:	FAILED error and POWER FAIL error if the transmit voltage is too low. This condition can be caused by a bad cable coupler.
Returns:	FAILED error if a DPSK Exclusive is in process.

Example:	`IMM>captureline`
	`<Error type='FAILED' msg='LINE BUSY'/>`
	`<Executed/>`

Example:	`IMM>captureline`
	`<Error type='FAILED' msg='Low Transmit Voltage - low battery or bad coupler'/>`
	`<Error type='POWER FAIL' msg='Transmit Voltage Vtx=1.9 '/>`
	`<Executed/>`

Data	Return host reply of last GData command (legacy). Use GetReply in new designs.		
Allowed from / Format:	Host? No	IM? Yes !00data	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	Contents of the GData reply buffer, prefixed with the instrument ID.		

Example:	`!01DATA`
	`<RemoteReply>01,this is the gdata reply</RemoteReply>`
	`<Executed/>`		

Disc (provisional)	Execute the line discovery process. This is a provisional command, the format and/or response may change upon further testing and development.		
Allowed from / Format:	Host? Yes disc	IM? No	Group? No
Modifies:	Discovered by list of all IMMs on the line		
Requires:	The IM line must be captured.		
Effects:	Uses the line discovery proves to generate a list of all IMMs on the mooring (except the one executing the command).		
Returns:	NOT ALLOWED error if the IM line is not captured.		

Example:	`IMM>disc`
	`<Discovery>`
	`<Discovered SN='7000020'/>`
	`<Discovered SN='7000004'/>`
	`</Discovery>`
	`<Executed/>`		

EchoIM:	IM communications echo test		
Allowed from / Format:	Host? No	IM? Yes !01echoim:text to echo	Group? No
Argument:	String to be echoed to the IM line		
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Echoes the string back to the IM line		

Example:	`!01echoim:this is a test`
	`<RemoteReply>echoim:this is a test</RemoteReply>`
	`<Executed/>`		

Exclusive:	Requests devices other than the target to go to sleep for 2 minutes Use this command when transmitting from a SIM to an IMM.		
Allowed from / Format:	Host? no use SENDEXCLUSIVE instead	IM? Yes exclusive:S70000019 exclusive:01	Group? no
Argument:	Device ID or serial number (prefaced with S) of target		
Modifies:	DPSK Transceiver state		
Requires:	nothing		
Effects:	causes all IMM's on the IM line other than the addressed device to enter a low power mode for two minutes. This saves power in applications with long data transfers.		

Example:	`S>exclusive:01`
	`EXCLUSIVE:01 ACCEPTED`
	`S>`		

ForceCaptureLine or FCL	Reserves IM line by transmitting a carrier signal, without first checking that no other device is using the IM line.		
Allowed from / Format:	Host? Yes forcecaptureline	IM? No	Group? No
Modifies:	DPSK Transceiver State		
Requires:	Nothing		
Effects:	Starts transmitting a carrier on the IM line without first determining if another device is already transmitting.		
Returns:	FAILED Error and POWER FAIL Error if the transmit voltage is too low. This condition can be caused by a bad cable coupler.		
Example:	`forcecaptureline` `<Executed/>`		

GData	Global command causes all IMMs to send value of GDataStr to their hosts. (**legacy**) Use group commands when possible in new designs: !G0:GDATA instead of GDATA		
Allowed from / Format:	Host? No	IM? Yes gdata	Group? Yes #G1:gdata
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	Global or group command – no return allowed.		

GetCD	Get Configuration Data		
Allowed from / Format:	Host? Yes getcd	IM? Yes !00getcd	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		

Returns:	All parameters in Configuration Data in the format:
	`<ConfigurationData DeviceType='SBE90554 IMM' SerialNumber='70000047'>`
	`<Settings ConfigType='2'`
	`DebugLevel='2'`
	`BaudRate='9600'`
	`HostID='Host ID not set'`
	`GdataStr='GDATA'`
	`HostPrompt='x'`
	`ModemPrompt='imm>'`
	`DeviceID='1'`
	`EnableHostFlagWakeup='0'`
	`EnableHostFlagConfirm='0'`		

	EnableHostFlagTerm='0' EnableSerialIMMWakeup='1' EnableHostPromptConfirm='1' EnableHostServeOnPwrup='1' EnableAutoIMFlag='1' EnablePrompt='1' EnableHostWakeupCR='1' EnableHostWakeupBreak='0' EnableEcho='1' EnableSignalDetector='1' EnableToneDetect='0' EnableFullPwrTX='0' EnableBackSpace='1' EnableGDataToSample='0' EnableStripHostEcho='0' EnableBinaryData='1' SerialType='1' TermToHost='254' TermFromHost='254' SerialBreakLen='5' MaxNumSamples='40' GroupNumber='3' THOST0='0' THOST1='5' THOST2='1000' THOST3='12000' THOST4='500' THOST5='5' TMODEM2='500' TMODEM3='18000' TMODEM4='100'/> </ConfigurationData> <Executed/>

GetConfigType	Get Configuration Type (1=SIM Compatibility, 2 = Standard IMM)		
Allowed from / Format:	Host? Yes getconfigtype	IM? Yes !00getconfigtype	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	<ConfigType T='2'/>		
Returns:	<ConfigType T='1'/>		

GetEC	Get Event Counter values (nonzero entries only)		
Allowed from / Format:	Host? Yes getec	IM? Yes !00getec	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	IMM>getec <EventSummary numEvents='187'/> <EventList DeviceType='SBE90554 IMM' SerialNumber='70000047'> <Event type='PowerOnReset' Count='1'/> <Event type='IMUartLockup' Count='186'/> </EventList> <Executed/>		

GetHD	Get Hardware Data

Allowed from / Format:	Host? Yes gethd	IM? Yes !00gethd	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	IMM>gethd <HardwareData DeviceType='SBE90554 IMM' SerialNumber='70000047'> <Manufacturer>Sea-Bird Electronics, Inc</Manufacturer> <HardwareVersion>41420B</HardwareVersion> <HardwareVersion>PCB Type 2, 10345A</HardwareVersion> <MfgDate>May 4 2006</MfgDate> <FirmwareVersion>IMM Ver 1.02</FirmwareVersion> <FirmwareDate>May 4 2006</FirmwareDate> </HardwareData> <Executed/> Definition of the lines: • **SerialNumber** - Each IMM and IM-enabled instrument has a unique serial number greater than 100 and less than 2^{32}. Serial numbers are set by Sea-Bird and cannot be changed. In Sea-Bird IM-enabled instruments, the serial number of the inductive modem is the serial number of the instrument. • **Manufacturer** - Always reads *Sea-Bird Electronics, Inc*. • **HardwareVersion** - A string identifying the assembly number of the IMM, or of the PCB which holds the IM circuit in a Sea-Bird IM-enabled instrument; at least 4 characters. • **MfgDate** - A string identifying the manufacture date; at least 6 characters. • **FirmwareVersion** - A string identifying the code version; limited to 16 characters. • **Firmwar Date** - A string identifying the compilation date; limited to 11 characters.		

GetHostFlag	Get HostFlag Signal value		
Allowed from / Format:	Host? Yes gethostflag	IM? Yes !00gethostflag	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	<HostFlag F='0'/>		
Returns:	<HostFlag F='1'/>		

GetHostID	Get HostID String		
Allowed from / Format:	Host? Yes gethostid	IM? Yes !00gethostid	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	<HostID>Host ID not set</HostID>		

GetLineStatus	Get status of the IM line (captured or idle)		
Allowed from / Format:	Host? Yes getlinestatus	IM? No	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	<LineStatus S='CAPTURED'/>		
Returns:	<LineStatus S='IDLE'/>		

GetReply	Get the host response to the last GData command		
Allowed from / Format:	Host? No	IM? Yes !00getreply	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	<GDataReply>the gdata reply</GDataReply>		
Example:	IMM>!00getreply <RemoteReply><GDataReply>Not Initialized </GDataReply> <Executed/> </RemoteReply> <Executed/>`		

GetSD	Get Status Data		
Allowed from / Format:	Host? Yes getsd	IM? Yes !00getsd	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	IMM>!01getsd <RemoteReply><Executing/> <StatusData DeviceType='SBE90554 IMM' SerialNumber='70000047'> <HostID>Host ID not set</HostID> <EventSummary numEvents='208'/> <Power><TransmitVoltage>7.6</TransmitVoltage></Power> <SampleDataSummary NumSamples='0' TotalLen='0' FreeMem='16384'/> <HostFileSummary Len='0' CRC='0x00000000'/> <DiscoveredBy SN='0'> <LineStatus>CAPTURED</LineStatus> </StatusData> <Executed/> </RemoteReply> <Executed/> Definition of the lines: • **SerialNumber** – IMM serial number (> 100 and < 2^{32}), programmed at factory. • **HostID** – User-programmable 4-64 character string. • **EventSummary** - Record of number of times events such as power loss, watchdog timer restarts, invalid commands, serial errors, timeouts, firmware failures, etc. have occurred. The generic data contains a vector of pre-defined integer error codes and the number of occurrences for each type. • **Power** - Battery voltage or other indication of the state of the device power supply. This may not be relevant when the IMM is powered through the host. • **SampleDataSummary** – Number of samples in memory, total amount of data in memory, and available memory in Sample Data • **HostFileSummary** – Total amount of data in memory and checksum for Host Data File • **DiscoveredBy** - Serial number of device that has identified this IMM in a line discovery process. There may be multiple instances of this tag. The tag may be removed in future firmware revisions. • **LineStatus** – Captured or idle		

HostFileAppendLine:	Append line to Host Data File		
Allowed from / Format:	Host? Yes hostfileappendline:data	IM? Yes !00hostfileappendline:data	Group? Yes !G1:hostfileappendline:data
Argument:	Data to be added to the Host Data file		
Modifies:	Host Data File		
Requires:	Nothing		
Effects:	Adds data following the colon to the end of the Host Data File. The amount of data stored with this command is limited by the length of the Command Buffer.		
Returns:	OVERFLOW Error if insufficient memory is available to store the data.		
Example:	`IMM>hostfileappendline:this is a test` `<Executed/>`		

HostFileErase	Erases the contents of the Host Data File		
Allowed from / Format:	Host? Yes hostfileerase	IM? Yes !00hostfileerase	Group? No
Modifies:	Host Data File		
Requires:	Nothing		
Effects:	Erases data from the Host Data File.		
Returns:	<ConfirmationRequired>		

Example:	`IMM>hostfileerase`
	`<WARNING>`
	`<Contents of HostFile will be erased!>`
	`</WARNING>`
	`<ConfirmationRequired/>`
	`<Executed/>`
	`IMM>hostfileerase`
	`<Executed/>`		

HostFileGetData	Retrieves the Host Data File		
Allowed from / Format:	Host? Yes hostfilegetdata	IM? Yes !00hostfilegetdata	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	The Host Data File, enclosed in a <HostData> tag.		

Example:	`IMM>hostfilegetdata`
	`<HostData Len='14' CRC='0x00000000'>this is a test</HostData>`
	`<Executed/>`		

HostFileGetSummary	Returns a summary of the Host Data File		
Allowed from / Format:	Host? Yes hostfilegetsummary	IM? Yes !00hostfilegetsummary	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	A <HostFileSummary> tag		

Example:	`IMM>hostfilegetsummary`
	`<HostFileSummary Len='14' CRC='0x00000000'/>`
	`<Executed/>`		

HostFileWrite	Writes binary data to the Host Data file		
Allowed from / Format:	Host? Yes hostfilewrite	IM? No	Group? No

Modifies:	Nothing
Requires:	Nothing
Effects:	Waits up to 3 seconds from the <Executing/> tag for the first data byte before timing out. Writes data to the Hostfile until a 1-second inter-character delay occurs, or the Host File overflows.
Returns:	<Executing/> when ready to receive data
Returns:	NOT ALLOWED Error if the Hostfile is not empty. The Hostfile must be erased prior to writing data.
Returns:	TIMEOUT Error if no bytes arrive within 3 seconds of the Executing tag.
Returns:	OVERFLOW Error if more than 4096 bytes arrive.

Example:	`IMM>hostfilewrite`
	`<Executing/>`
	`this is a test`
	`<Executed/>`

Example:	`IMM>hostfilewrite`
	`<Error type='NOT ALLOWED' msg='Erase Hostfile before writing'/>`
	`<Executed/>`

ID?	Returns Device ID (**legacy**) Use GetCD or Disc in new designs.		
Allowed from / Format:	Host? Yes[37] (Captured Line only) id?	IM? Yes id?	Group? No
This is a global command that returns data. It must not be used if more than one instrument is coupled to the IM line. This command does not use reply tags.			
Modifies:	Nothing		
Requires:	There must be exactly one receiving IM device on the mooring.		
Effects:	Returns data		
Returns:	The Device ID of the IM device on the line		

Example:	`IMM>id?`
	`<RemoteReply>id = 00`
	`</RemoteReply><Executed/>`		

MeasureLineSignal or MLS	Measures the incoming DPSK signal on the IM line		
Allowed from / Format:	Host? Yes measurelinesignal	IM? No	Group? No
Modifies:	DPSK Transceiver state		
Requires:	Nothing		
Effects:	Echoes the string back to the IM line		
Returns:	<LineSignal S='48'/> <Executed/> Note: Typical value is 41-43 when device is transmitting to IM line, 47-49 when a wake-up tone is present. Value <38 or >50 indicates no signal, noise, or weak signal.		

Example:	`IMM>measurelinesignal`
	`<LineSignal S='42'/>`
	`<Executed/>`		

PwrOff	Tell IMM to go to sleep.		
Allowed from / Format:	Host? Yes pwroff	IM? Yes !00pwroff	Group? Yes !G1:pwroff
Modifies:	IMM operating mode		

[37] When the Host sends the ID? command to the IMM, the IMMsends the global ID? to the IM line. If an ID is returned it is the ID of a remote device, not the IMM connected serially to the Host.

Requires:	Nothing
Effects:	Unconditionally switches the IMM from Host Service or IM Service to Sleep mode after a 100mS delay. When received from the Host, sends the global IM command PwrOff to the IM line if the line is captured before powering off.
Returns:	<Executed/>
Example: (line captured)	IMM>pwroff <Executing/> <Executed/>
Example: (line not captured)	IMM>pwroff <Executed/>

ReleaseLine or REL	Releases the IM line, allowing other devices to use it		
Allowed from / Format:	Host? Yes releaseline	IM? No	Group? No
Modifies:	DPSK Transceiver State		
Requires:	Nothing		
Effects:	Stops transmitting the IM carrier. Automatically sends PWROFF to the IM line before releasing the line.		
Returns:	<Executed/>		

ResetEC	Resets all event counters to zero		
Allowed from / Format:	Host? Yes resetec	IM? Yes !00resetec	Group? Yes !G1:resetec
Modifies:	Event Counters		
Requires:	Nothing		
Effects:	Resets all event counters to zero		
Returns:	<Executed/>		

ResetEventCounters	Same as ResetEC. See ResetEC above.

SampleAdd	Creates a new sample in Sample Data and adds binary data to it.		
Allowed from / Format:	Host? Yes sampleadd	IM? No	Group? No
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Creates a new sample, then sends <Executing/> tag to host. Adds binary data from the host to the new sample until a 1-second inter-character delay occurs or sample memory overflows.		
Returns:	<Executing/> when ready to receive data		
Returns:	TIMEOUT Error if no bytes arrive within 3 seconds of the Executing tag.		
Returns:	OVERFLOW Error if more bytes arrive than can be stored in Sample Data		
Returns:	WARNING tag if the Sample data structure contained MaxNumSamples[38] before executing this command. The warning indicates that the oldest Sample in memory was deleted to make room for the new one. User software should avoid dependencies on the contents of Warning tags.		
Example:	IMM>sampleadd <Executing/> this is a test <Executed/>		

[38] MaxNumSamples is part of Configuration Data, settable to any value from 1 to 40.

SampleAddLine:	Creates a new sample in Sample Data and adds data to it		
Allowed from / Format:	Host? Yes sampleaddline:data	IM? Yes !00sampleaddline:data	Group? Yes !G1:sampleaddline:data
Argument:	Data to write to the new sample (not binary, CRLF termination is no added to the sample)		
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Creates a new sample and adds data following the colon to it. The amount of data stored with this command is limited by the length of the Command Buffer		
Returns:	OVERFLOW Error if more bytes arrive than can be stored in Sample Data		
Returns:	WARNING tag if the Sample data structure contained MaxNumSamples before executing this command. The warning indicates that the oldest Sample in memory was deleted to make room for the new one. User software should avoid dependencies on the contents of Warning tags.		
Example:	`IMM>sampleaddline:this is a test` `<Executed/>`		

SampleAppend	Appends binary data to current sample in Sample Data		
Allowed from / Format:	Host? Yes sampleappend	IM? No	Group? No
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Creates a new Sample if there are none. Sends <Executing/> tag to host. Appends binary data from the host to the current Sample until a 1-second inter-character delay occurs or sample memory overflows.		
Returns:	<Executing/> when ready to accept data		
Returns:	TIMEOUT Error if no bytes arrive within 3 seconds of the Executing tag.		
Returns:	OVERFLOW Error if more bytes arrive than can be stored in Sample Data		
Returns	<Executed/> when data entry terminated		

SampleAppendLine:	Append data to current sample in Sample data		
Allowed from / Format:	Host? Yes sampleappendline:data	IM? Yes !00sampleappendline:data	Group? Yes !G1:sampleappendline:data
Argument:	data to append (not binary, CRLF terminatin is not added to sample)		
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Adds data following the colon to the current sample. The amount of data stored with this command is limited by the length of the Command Buffer		
Returns:	OVERFLOW Error if more bytes arrive than can be stored in Sample Data		
Example:	`IMM>sampleappendline:this is a test` `<Executed/>`		

SampleErase:	Erases a single Sample with specified SampleID; sample must be the oldest sample in Sample Data		
Allowed from / Format:	Host? Yes sampleerase:FF	IM? Yes !00sampleerase:FF	Group? No
Argument:	A hex SampleID must follow the colon (up to 8 characters for a 32 bit number). The sample ID of hte sample to be erased must be specified to avoid accidental deletion of samples.		
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Erases the Sample with specified SampleID if it is the oldest sample in the Sample Data structure.		
Returns:	INVALID ARGUMENT Error if there is not a properly formatted SampleID after the colon.		
Returns:	INVALID ARGUMENT Error if the specified SampleID does not exist.		
Returns:	<Executed/>		

SampleEraseAll	Unconditionally erases all samples in Sample Data		
Allowed from / Format:	Host? Yes sampleeraseall	IM? Yes !00sampleeraseall	Group? Yes !G1:sampleeraseall
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Erases all Samples in Sample memory.		
Returns:	<Executed/>		

SampleEraseMultiple:	Erases multiple samples, starting with the specified SampleID and including all lower SampleIDs		
Allowed from / Format:	Host? Yes sampleerasemultiple:FF	IM? Yes !00sampleerasemultiple:FF	Group? No
Argument:	A 32-bit hex SampleID must follow the colon.		
Modifies:	Sample Data		
Requires:	Nothing		
Effects:	Erases the Sample with specified SampleID, and all Samples in Sample Data with lower Sample IDs.		
Returns:	INVALID ARGUMENT Error if there is not a properly formatted SampleID after the colon.		
Returns:	INVALID ARGUMENT Error if the specified SampleID does not exist.		
Returns:	<Executed/>		

SampleGetData:	Retrieves the data of a specified sample in Sample Data		
Allowed from / Format:	Host? Yes samplegetdata:FF	IM? Yes !00samplegetdata:FF	Group? No
Argument:	A 32-bit hex SampleID must follow the colon.		
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	INVALID ARGUMENT Error if there is not a properly formatted SampleID after the colon.		
Returns:	INVALID ARGUMENT Error if the specified SampleID does not exist.		
Returns:	The data for the specified Sample in a SampleData tag		

Example:	`IMM>samplegetdata:22`
	`<SampleData ID='0x22' LEN='14' CRC='0x0'>this is a test</SampleData>`
	`<Executed/>`		

SampleGetLast	Retrieves the data of the most recent sample		
Allowed from / Format:	Host? Yes samplegetlast	IM? Yes !00samplegetlast	Group? No
Argument:	none		
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	INVALID ARGUMENT Error if there are no samples in memory		
Returns:	The data for the most recent sample in a SampleData tag		

Example:	`IMM>samplegetlast`
	`<SampleData ID='0x22' LEN='14' CRC='0x0'>this is a test</SampleData>`
	`<Executed/>`		

SampleGetList	Retrieves a list of samples in Sample Data		
Allowed from / Format:	Host? Yes samplegetlist	IM? Yes !00samplegetlist	Group? No

Modifies:	Nothing
Requires:	Nothing
Effects:	Returns data
Returns:	A SampleList tag with entries (sample ID, sample size, and check sum) for each sample in Sample Data.
Example:	IMM>samplegetlist <SampleList> <Sample ID='0x00000022' Len='14' CRC='0x00000000'/> <Sample ID='0x00000021' Len='14' CRC='0x00000000'/> </SampleList> <Executed/>

SampleGetSummary	Retrieves a summary of the data in Sample Data		
Allowed from / Format:	Host? Yes samplegetsummary	IM? Yes !00samplegetsummary	Group? No
Modifies:	Nothing		
Requires:	Nothing		
Effects:	Returns data		
Returns:	A summary of the data in Sample Data in a SampleDataSummary tag		
Example:	IMM>samplegetsummary <SampleDataSummary NumSamples='2' TotalLen='28' FreeMem='16356'/> <Executed/>		

SendBreak	Sends a serial break to the host.		
Allowed from / Format:	Host? No	IM? Yes !00sendbreak	Group? Yes !G1:sendbreak
Modifies:	Host Serial interface		
Requires:	Nothing		
Effects:	Sends a serial break to the host		
Returns:	<Executed/>		

SendExclusive:	Requests devices other than the target to go to sleep for 2 minutes (use EXCLUSIVE command when transmitting from a SIM)		
Allowed from / Format:	Host? Yes sendexclusive:01	IM? no	Group? no
Argument:	Device ID or serial number (prefaced with S) of target		
Modifies:	DPSK Transceiver state		
Requires:	nothing		
Effects:	Causes all IMM's on the IM line other than the addressed device to enter Sleep mode for two minutes. This saves power in applications with long data transfers.		
Example:	IMM>sendexclusive:07 <RemoteReply>EXCLUSIVE:07 ACCEPTED </RemoteReply> <Executed/>		
Example:	IMM>sendexclusive:S70000047 <RemoteReply>EXCLUSIVE:S70000047 ACCEPTED </RemoteReply> <Executed/>		

SendGData	Sends the global IM command GData to the IM line		
Allowed from / Format:	Host? Yes (Captured Line only) sendgdata	IM? No	Group? No
Requires:	IM line must be captured		

Modifies:	IM line
Effects:	Sends the global IM command GData to the IM line
Returns:	NOT ALLOWED Error if the IM line is not captured.
Returns:	<Executing/> <Executed/>

SendPwrOff	Sends the global IM command PwrOff to the IM line		
Allowed from / Format:	Host? Yes (Captured Line only) sendpwroff	IM? No	Group? No
Requires:	IM line must be captured		
Modifies:	IM line		
Effects:	Sends the global IM command PwrOff to the IM line; IMM communicating with the host remains awake and line remains captured.		
Returns:	NOT ALLOWED Error if the IM line is not captured.		
Returns:	<Executing/> <Executed/>		

SendStayOn	Sends the global IM command StayOn to the IM line, to reset the 2-minute sleep timer for all IMMs on the line. IMM communicating with the host is not affected.		
Allowed from / Format:	Host? Yes (Captured Line only) sendstayon	IM? No	Group? No
Requires:	IM line must be captured		
Modifies:	IM line		
Effects:	Sends the global IM command StayOn to the IM line		
Returns:	NOT ALLOWED Error if the IM line is not captured.		
Returns:	<Executing/> <Executed/>		

SendWakeupTone or SWT	Sends a 4-second wake-up tone to the IM line.		
Allowed from / Format:	Host? Yes (Captured Line only) sendwakeuptone	IM? No	Group? No
Requires:	IM line must be captured		
Modifies:	IM line		
Effects:	Sends a 4-second wake-up tone to the IM line		
Returns:	NOT ALLOWED Error if the IM line is not captured.		
Returns	<Executing/> about once per second, followed by <Executed/> when finished.		

Example:	`IMM>sendwakeuptone`
	`<Executing/>`
	`<Executing/>`
	`<Executing/>`
	`<Executed/>`		

Set commands	See *Configuration Type 2 Set Commands* below.

StayOn	Resets the IMM's 2-minute sleep timer		
Allowed from / Format:	Host? Yes stayon	IM? Yes !00stayon	Group? Yes !G1:stayon
Requires:	Nothing		
Modifies:	2-minute sleep timer		

Effects:	Resets the IMM's 2 minute sleep timer, keeping the device awake.

Test20TurnCoupler or T20CC	Tests the integrity of the IM cable coupler. **For 20-turn couplers only.**		
Allowed from / Format:	Host? Yes (Captured Line only) test20turncoupler	IM? Yes !00test20turncoupler	Group? No
Requires:	IM line must be captured if command sent from host		
Requires:	This command requires a 20-turn coupler, or test results will be misleading.		
Modifies:	IM line		
Effects:	Performs a test of the IM cable coupler		
NOTE:	When the test20turncoupler command is received from the IM line, several characters of noise will be transmitted while the test is in-process. See the example in the TestCableCoupler Command description below.. This noise is an unavoidable result of the method of testing the coupler. The noise does not occur when the command is received in Host Service mode.		
Returns:	NOT ALLOWED Error if the IM line is not captured when command is received from the host.		
Returns:	FAILED Error and POWER FAIL Error if the transmit voltage is low.		
Returns:	FAILED Error if the coupler connection is poor or the coupler is not connected.		
Returns:	<CableCouplerTest status='GOOD | MEDIUM | BAD' Z='UINT16'/> if the test is performed		

TestCableCoupler or TCC	Tests the integrity of the IM cable coupler. **For 100-turn couplers only.**		
Allowed from / Format:	Host? Yes (Captured Line only) testcablecoupler	IM? Yes !00testcablecoupler	Group? No
Requires:	IM line must be captured if command sent from host		
Requires:	This command requires a 100-turn coupler, or test results will be misleading.		
Modifies:	IM line		
Effects:	Performs a test of the IM cable coupler		
NOTE:	When the testcablecoupler command is received from the IM line, several characters of noise will be transmitted while the test is in-process. See the example below. This noise is an unavoidable result of the method of testing the coupler. The noise does not occur when the command is received in Host Service mode.		
Returns:	NOT ALLOWED Error if the IM line is not captured when command is received from the host.		
Returns:	FAILED Error and POWER FAIL Error if the transmit voltage is low.		
Returns:	FAILED Error if the coupler connection is poor or the coupler is not connected.		
Returns:	<CableCouplerTest status='GOOD | MEDIUM | BAD' Z='UINT16'/> if the test is performed		

Example:	`IMM>testcablecoupler`
	`<CableCouplerTest status='GOOD' Z='1485'/>`
	`<Executed/>`		

Example:	`IMM>testcablecoupler`
	`<Error type='NOT ALLOWED' msg='IM Line Not Captured'/>`
	`<Executed/>`		

Example:	`IMM>testcablecoupler`
	`<Error type='FAILED' msg='Low Transmit Voltage - low battery or bad coupler'/>`
	`<Error type='POWER FAIL' msg='Transmit Voltage Vtx=0.0 '/>`
	`IMM>`		

Example	`!00TestCAbleCoupler`
	`<RemoteReply><Executing/>Ðÿ!#Ãÿÿyàÿÿo¿ÿ„ÿ`
	`àÿ<CableCouplerTest status='GOOD' Z='2405'/>`
	`<Executed/>`
	`</RemoteReply>`
	`<Executed/>`		

TestEEPROM	Tests the internal EEPROM for errors.

Allowed from / Format:	Host? Yes testeeprom	IM? Yes !00testeeprom	Group? No
Requires:	Nothing		
Modifies:	Nothing		
Effects:	Tests the internal EEPROM for errors.		
Returns:	<Executing/> about once per second.		
Returns:	One or more EEPROM FAILURE Errors if the test does not pass		
Returns (when completed):	<Executed/>		

TXTest	Sends a series of tests of IM communications		
Allowed from / Format:	Host? No	IM? Yes !00txtest	Group? No
Requires:	Nothing		
Modifies:	Nothing		
Effects:	Sends 10 seconds of silence, followed by 250 *U* characters at maximum speed, followed by 250 *U* characters at pseudo-random intervals.		

11.6 Config Type 2 Set Commands

In Config Type 2, all commands that change the Configuration Data begin with the keyword *Set* and end with = followed by some argument. All *Set* command arguments are tested for minimum and maximum values. If the argument is a string, it is tested for minimum and maximum length. If the command contains the keyword *Enable* then the argument must be 1 or 0. If the argument is not valid, the IMM replies with *INVALID ARGUMENT Error*. Some parameters respond with the <ConfirmationRequired/> tag, indicating the command must be repeated to take effect. The IMM always sends the <Executed/> tag, even when the parameter is not changed; the Executed tag indicates the processing of the command is complete – it does not indicate that the command had any effect.

SetBaudRate=x
x= Baud rate for host communication (1200, 2400, 4800, 9600, 19200[39]). Returns ConfirmationRequired tag and Warning tag. The IMM goes to sleep after this command is accepted. When it is woken up again, it uses the new baud rate.
SetBaudRate is allowed from the host only.
Factory default is 9600. Baud rate is not affected by a change in Interface mode.

SetConfigType=x
x= 1: Configuration Type 1 (SIM Compatibility).
x= 2: Configuration Type 2 (Standard 2005).
Returns ConfirmationRequired tag and Warning tag. This command should not be used after deployment.
SetConfigType is allowed from the host only.
Factory default is 2 (Config Type 2). Config Type is not affected by a change in Interface mode.

SetDebugLevel=x
x= 0: Disable all executed and executing tags, and remote reply tags.
x= 1: Disable all remote reply tags.
x= 2: Normal operation.
x= 3: For debugging with a terminal program. This enables some extra text output <!--in XML comment tags--> in response to many commands. The existence and content of comment tags is not part of the IMM specification - IMM applications should have no dependency on the existence or content of XML comments.
x=4 – 9: Factory use only.
SetDebugLevel is allowed from the host, the IM, and as a group command.
Factory default is DebugLevel=2 (normal operation). DebugLevel is not affected by a change in Interface mode.
Note: THost4 timeouts are reported as Errors only if DebugLevel is set to 6 or higher.

SetDeviceID=x
x= Device ID (00 – 99), which is the *ii* in !iiCOMMAND or #iiCOMMAND.
SetDeviceID is allowed from the host and the IM.
Factory default is 0. Device ID is not affected by a change in Interface mode. Two digits are required for #ii or !ii commands, but the SetDeviceID command does not require two digits. SETDEVICEID=01 has the same effect as SETDEVICEID=1, !00GETSD gets the status data of remote device with ID =00 and !0GETSD returns an error

SetEnableAutoIMFlag=x
x= 1: IMM automatically controls the IMFlag digital output to wake up the host, confirm the IMM wakeup, and terminate commands and replies.
x= 0: Disable this feature. When set to 0, the only way to change the state of IMFlag is with the SetIMFlag= command.
SetEnableAutoIMFlag is allowed from the host, the IM, and as a group command.
Default is 1 for all Interface modes.

[39] Support for 38400 baud was removed in firmware V1.05

SetEnableBackSpace=x
x= 1: Interpret the backspace character as a command to delete the previously received character when received from the host. This feature applies only to host commands, not to host replies to IM commands. This feature is extremely useful when the host is a terminal program on a PC.
x= 0: Treat the backspace character as a normal character with no deletion.
SetEnableBackSpace is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1 through 6 and 8 through 13; default is 1 for Interface modes 7 and 14.

SetEnableBinaryData=x
x= 1: Enable the IMM-specific translation of data to the IM line.
x= 0: Disable IMM specific translation. Use this setting when using the IMM with older devices that do not support translation, such as the SBE 44, UIM, and SIM. Older devices will not properly receive < > : \n * @ ! and # if binary data is enabled on the transmitting IMM.
SetEnableBinaryData is allowed from the host, the IM, and as a group command.
Default is 1 for all Interface modes.

SetEnableEcho=x
x= 1: IMM echoes all characters received from the host back to the host.
x= 0: Disable the echo.
SetEnableEcho is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1 through 6 and 8 through 13; default is 1 for Interface modes 7 and 14.

SetEnableFullPwrTX=x
x= 1: IMM transmits with the lesser of 9V or the input voltage. The higher power transmit option increases signal strength by as much as a factor of 3, and increases power consumption during transmission to 50mW at 10V (with 100-turn coupler).
x= 0: Use low power transmit mode with transmitting power consumption of 10mW at10V.
SetEnableFullPwrTX is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

SetEnableGDataToSample=x
x= 1: IMM stores host response to GData command as a sample in Sample Data instead of storing the response in the GData Reply Buffer. This removes the 127 byte limitation on the host reply.
x= 0: IMM Stores host response to GData command in the 127 byte GData Reply Buffer.
SetEnableGDataToSample is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

SetEnableHostFlagConfirm=x
x= 1: IMM interprets a low-high transition on the HostFlag signal to confirm wake-up of the host.
x= 0: Disable this confirmation.
SetEnableHostFlagConfirm is allowed from the host and the IM.
Default is 0 for Interface modes 2, 4, 6, 7, 9, 11, 13, and 14; default is 1 for Interface modes 1, 3, 5, 8, 10, and 12.

SetEnableHostFlagTerm=x
x= 1: IMM interprets a high-low transition on the HostFlag signal to terminate a transmission from the host.
x= 0: Disable this termination.
SetEnableHostFlagTerm is allowed from the host and the IM.
Default is 0 for Interface modes 2, 4, 6, 7, 9, 11, 13, and 14; default is 1 for Interface modes 1, 3, 5, 8, 10, and 12.

SetEnableHostFlagWakeup=x
x= 1: IMM wakes up in response to low-high transitions on the HostFlag digital input. Also, IMM wakes up in response to a static logic high condition on the HostFlag signal when it is about to enter sleep, including after receiving commands from the IM line with non-matching address.
x= 0: Disable this wake up.
SetEnableHostFlagTerm is allowed from the host and the IM.
Default is 0 for Interface modes 1, 2, 4, 6, 7, 8, 9, 11, 13, and 14; default is 1 for Interface modes 3, 5, 10, and 12.

SetEnableHostPromptConfirm=x
x= 1: IMM interprets reception of a prompt from the host as confirmation of the host wake-up.
x= 0: Disable this confirmation.
SetEnableHostPromptConfirm is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1 through 6 and 8 through 13; default is 1 for Interface modes 7 and 14.

SetEnableHostServeOnPwrUp=x
x= 1: IMM enters Host Service mode when power is turned on (or when the IMM resets for any other reason).
x= 0: Disable this feature.
SetEnableHostServeOnPwrUp is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1, 2, 5, 6, 8, 9, 12, and 13;default is 1 for Interface modes 3, 4, 7, 10, 11, and 14.

SetEnablePrompt=x
x= 1: IMM sends the ModemPrompt to the host when it powers up, when it wakes up in response to HostFlag or Host Serial transitions, after processing each host command (except for the PwrOff command), and after confirming host wake-up when relaying IM commands.
x= 0: Disable this prompt.
SetEnablePrompt is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1 through 6 and 8 through 13; default is 1 for Interface modes 7 and 14.

SetEnableHostWakeupCR=x
x= 1: IMM sends a character to the host to wake-up the host when a command is received from the IM line. The IMM sends a '\r' to the host to generate the wakeup. This feature is independent of IMFlag setttings.
x= 0: Disable this feature.
SetEnableHostWakeupCR is allowed from the host, the IM, and as a group command.
Default is 0 for Interface modes 1 through 5 and 8 through 12; default is 1 for Interface modes 6, 7, 13, and 14.

SetEnableHostWakeupBreak=x
x= 1: IMM sends a serial break condition to the host to wake-up the host when a command is received from the IM line. The SerialBreakLen setting controls the duration of the break condition. This feature is independent of IMFlag setttings.
x= 0: Disable this feature.
SetEnableHostWakeupBreak is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

SetEnableSerialIMMWakeup=x
x= 1: IMM wakes up in response to high-low transitions on the serial input.
x= 0: Disable this wake up.
Returns ConfirmationRequired tag and Warning tag if set to 0.
SetEnableSerialIMMWakeup is allowed from the host and the IM.
Default is 1 for all Interface modes.
Note: EnableSerialIMMWakeup cannot be set to 0 if EnableSignalDetector is set to 0. This prevents the IMM from reaching a state from which it cannot wake up.

SetEnableSignalDetector=x
x= 1: Enable the IM signal detector. The IMM wakes up in response to commands on the IM line.
x= 0: Disable the signal detector.
Returns ConfirmationRequired tag and Warning tag if set to 0.
SetEnableSignalDetector is allowed from the host only.
Default is 1 for Interface modes 1, 2, 5, 6, 7, 8, 9, 12, 13, and 14; default is 0 for Interface modes 3, 4, 10, and 11.
Note: EnableSignalDetector cannot be set to 0 if EnableSerialIMMWakeup is set to 0. This prevents the IMM from reaching a state from which it cannot wake up.

SetEnableStripHostEcho=x
x= 1: The IMM in IM Service mode automatically removes the host's echo of the relayed command, if there is an echo.
x= 0: The IMM does not strip any characters from the host reply.
SetEnableStripHostEcho is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

SetEnableToneDetect=x
x= 1: IMM wakes the host when an incoming wake-up tone is detected.
x= 0: Disable IM notification of incoming wake-up tones.
SetEnableToneDetect is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

Example:
Wake-up tone is enabled in IMM1, IMM1 put to sleep.

```
IMM>setenabletonedetect=1
<Executed/>
IMM>pwroff
<Executed/>
```

Another IMM (2) sends a wake-up tone to the IM line:

```
IMM>sendwakeuptone
<Executing/>
<Executing/>
<Executing/>
<Executing/>
<Executed/>
```

While wake-up tone is sent from IMM2, IMM1 sends this to host:

```
IMM>WAKE-UP TONE DETECTED
IMM>WAKE-UP TONE DETECTED
IMM>WAKE-UP TONE DETECTED
[... continues to signal until tone stops...]
IMM>WAKE-UP TONE DETECTED
```

SetGDataStr=x
x= GDataStr value (1 - 32 characters), which is sent to the host in response to the GData command.
SetGDataStr is allowed from the host, the IM, and as a group command.
Factory default is GDATA. The GData string is not affected by a change in Interface mode.

SetGroupNumber=x
x= Group membership of the IMM (1 – 9, or 0). Group 0 is pre-defined as the group of all instruments; a 0 entry for this parameter indicates the IMM is not a member of any other group. Individual devices cannot belong to more than two groups – group 0 plus one additional group.
SetGroupNumber is allowed from the host, the IM, and as a group command.
Factory default is Group 0. Group membership is not affected by a change in Interface mode.

SetHostID=x
x= Host ID string (4 – 64 characters).
SetHostID is allowed from the host, the IM, and as a group command.
Default is *Host ID not set* for all Interface modes.

SetHostPrompt=x
x= Prompt string expected from the host to confirm wake-up (1 – 7 characters). Host prompts are not case sensitive. For example, if the hostprompt is set to *s*> and the instrument sends *S*>, the prompt is accepted.
SetHostPrompt is allowed from the host, the IM, and as a group command.
Default is S> for Interface modes 1 through 6 and 8 through 13; default is x for Interface modes 7 and 14.

SetID=x
Same as SetDeviceID=.

SetIMFlag=x
x=1: Deactivate the pulldown on the IMFlag output line.
x=0: Activate the pulldown on the IMFlag output line.
SetIMFlag is allowed from the host, the IM, and as a group command.
NOTE: this is not a setting in Configuration Data - this changes the state of the physical IO line. If ENABLEAUTOIMFLAG=1, the IMFlag signal will be released a few milliseconds after parsing the command.

SetInterfaceMode=x
Note: This command changes many parameters in Configuration Data. Check the desired Interface mode carefully before sending this command. Confirmation is required. The IMM goes to sleep after this command is accepted. When it is woken up again, it uses the new Interface mode.
The following parameters are not changed by the SetInterfaceMode command: ConfigType, DebugLevel, BaudRate, HostId, GDataStr, DeviceID, GroupMembership.
Returns ConfirmationRequired tag and Warning tag.
SetInterfaceMode is allowed from the host only.
Default Interface mode (from the factory) is 7.
Note: For Versions < 1.0, this command was SetConfigMode=.

Example:
```
IMM>setinterfacemode=7
<WARNING>
IMM will power down
next power up after confirm will use
new Configuration Data settings
with RS232 Serial interface
</WARNING>
<ConfirmationRequired/>
<Executed/>
IMM>setinterfacemode=7
<Executed/>
```

SetMaxNumSamples=x
x= Maximum number of samples allowed in Sample Data (1 – 40). If the maximum is set to a number smaller than the number of samples currently in memory, each SampleAdd or SampleAddLine deletes only one old sample before creating a new sample, leaving the number of samples in memory greater than the specified maximum.
SetMaxNumSamples is allowed from the host, the IM, and as a group command.
Default is 40 for all Interface modes.

SetModemPrompt=x
x= Prompt sent by the IMM to the host to cause a wake-up (1 – 7 characters).
SetModemPrompt is allowed from the host, the IM, and as a group command.
Default is IMM for all Interface modes.

SetSerialBreakLen=x
x= Length of a serial break in tens of milliseconds (1 – 255, corresponding to 10 milliseconds to 2.55 seconds). Not affected by the baud rate.
SetSerialBreakLen is allowed from the host, the IM, and as a group command.
Default is 5 for all Interface modes.

SetSerialType=x
x=0: Logic Level Serial
x=1: RS-232.
x=2: RS-485. Note that RS-485 requires a special hardware option.
Returns ConfirmationRequired tag and Warning tag.
SetSerialType is allowed from the host and the IM.
Default is 1 (RS-232) for Interface modes 1-7, zero for 8-14.

SetTermFromHost=x
x= Termination character for host replies to the IMM in IM Service mode (decimal number from 0 – 250).
TermFromHost=254 is carriage return line feed (‘\r\n’) and 255 is no termination character. 251, 252, and 253 are reserved. A serial break condition cannot be used for termination from the host. Note: TermFromHost does not apply in Host Service mode! In Host Service mode the termination sequence is always ‘\r\n’. Note that termination characters from the host are included in IM replies.
SetTermFromHost is allowed from the host, the IM, and as a group command.
Default is 254 for Interface modes 1, 3-8, and 10-14; default is 62(decimal) for Interface modes 2 and 9.

SetTermToHost=x
x= Termination character for transmissions from the IMM to the host (decimal number from 0 – 250). TermToHost =253 is a serial line break, 254 is carriage return line feed (‘\r\n’), and 255 is no termination character. 251 and 252 are reserved.
SetTermToHost is allowed from the host, the IM, and as a group command.
Default is 13 for Interface modes 1 through 6 and 8 through 13; default is 254 for Interface modes 7 and 14.

SetTHost0=x
x= THost0 maximum wait for wake-up confirmation (0 - 1000; units are tens of milliseconds; THost0=100 means 1 second). Described in detail in *Section 8.6:T imeout Parameters*.
SetTHost0 is allowed from the host, the IM, and as a group command.
Default is 0 for all Interface modes.

SetTHost1=x
x= THost1 delay after confirm (0 - 300; units are tens of milliseconds; THost1=100 means 1 second). Described in detail in *Section 8.6 Timeout Parameters in IM Service Mode*.
SetTHost1 is allowed from the host, the IM, and as a group command.
Default is 5 for all Interface modes.

SetTHost2=x
x= THost2 maximum wait for start of reply (0, or 100 - 3000; units are tens of milliseconds; THost2=100 means 1 second). Described in detail in *Section 8.6 Timeout Parameters in IM Service Mode*.
SetTHost2 is allowed from the host, the IM, and as a group command.
Default is 3000 for Interface modes 1, 3, 5, 8, 10, and 12; default is 1000 for Interface modes 2, 4, 6, 7, 9, 11, 13, and 14.

SetTHost3=x
x= THost3 IM reply transmission time (100 - 18000; units are tens of milliseconds; THost3=100 means 1 second). Described in detail in *Section 8.6 Timeout Parameters in IM Service Mode*.
SetTHost3 is allowed from the host, the IM, and as a group command.
Default is 12000 for all Interface modes.

SetTHost4=x
x= THost4 inter-character delay timeout (5 - 3000); units are tens of milliseconds; THost4=100 means 1 second). Described in detail in *Section 8.6 Timeout Parameters in IM Service Mode*.
SetTHost4 is allowed from the host, the IM, and as a group command.
Default is 500 for all Interface modes.

SetTHost5=x
x= THost5 IMFlag hold after reply termination. (5 - 3000); units are tens of milliseconds; THost4=100 means 1 second). Described in detail in *Section 8.6 Timeout Parameters in IM Service Mode*.
SetTHost5 is allowed from the host, the IM, and as a group command.
Default is 5 for all Interface modes.

SetTModem2=x
x= TModem2 maximum wait for start of IM reply (0 - 3000); units are tens of milliseconds; TModem2=100 means 1 second). Described in detail in *Section 5.5 Timeout Parameters in Host Service Mode*.
SetTModem2 is allowed from the host, the IM, and as a group command.
Default is 500 for all Interface modes.

SetTModem3=x
x= TModem3 maximum IM receive time (0 - 60000); units are tens of milliseconds; TModem3=100 means 1 second). Described in detail in *Section 5.5 Timeout Parameters in Host Service Mode*.
SetTModem3 is allowed from the host, the IM, and as a group command.
Default is 18000 for all Interface modes.

SetTModem4=x
x= TModem4 Configuration Type 1 maximum IM receive inter-character delay before terminating reception of reply to bii command (0 – 3000; units are tens of milliseconds; TModem4=100 means 1 second). Described in detail in *Section 5.5 Timeout Parameters in Host Service Mode*. This setting exists in IMM firmware V1.02 and later.
SetTModem4 is allowed from the host, the IM, and as a group command.
Default is 100 for all Interface modes.

12 Interface Mode Shortcuts

Interface modes act as shortcuts for common interface settings in Configuration Data. Setting an Interface mode changes the Configuration Data to support a particular style of host interface, as defined in the table below. There are currently only seven basic interface modes, but separate numbers are assigned for the same configuration in RS-232 and in LLS.

Interface Mode Number	Description	HostPrompt	ModemPrompt	EnableHostFlagWakeup	EnableHostFlagConfirm	EnableHostFlagTerm	EnableSerialMMWakeup	EnableHostPromptConfirm	EnableHostServeOnPwrup	EnableAutoIMFlag	EnablePrompt	EnableHostWakeupCR	EnableHostWakeupBreak	EnableEcho	EnableSignalDetector	EnableToneDetect	EnableFullPwrTX	EnableBackSpace	EnableGDataToSample	EnableStripHostEcho	EnableBinaryData	SerialType	TermToHost (decimal)	TermFromHost (decimal)	SerialBreakLen	MaxNumSamples	GroupNumber	THOST0	THOST1	THOST2	THOST3	THOST4	THOST5	TMODEM2	TMODEM3	TMODEM4
1	RS232 - Hardware Handshaking with Simple Host Instrument	S>	IMM>	0	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
2	RS232 - Serial Handshaking with Simple Host Instrument	S>	IMM>	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	13	62	5	40	0	0	5	1000	12000	500	5	500	18000	100
3	RS232 - Hardware Handshaking with Simple Buoy Controller	S>	IMM>	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
4	RS232 - Serial Handshaking with Simple Buoy Controller	S>	IMM>	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	13	254	5	40	0	0	5	1000	12000	500	5	500	18000	100
5	RS232 - Hardware Handshaking for Symmetric Communication	S>	IMM>	1	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
6	RS232 - Serial Handshaking for Symmetric Communication	S>	IMM>	0	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	0	1	1	13	254	5	40	0	0	5	1000	12000	500	5	500	18000	100
7 (default)	RS232 - Terminal Program Symmetric Communication	X>	IMM>	0	0	0	1	1	0	1	1	1	0	1	1	0	0	1	0	0	1	1	254	254	5	40	0	0	5	1000	12000	500	5	500	18000	100
8	Logic Level Serial - Hardware Handshaking with Simple Host Instrument	S>	IMM>	0	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
9	Logic Level Serial - Serial Handshaking with Simple Host Instrument	S>	IMM>	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	13	62	5	40	0	0	5	1000	12000	500	5	500	18000	100
10	Logic Level Serial - Hardware Handshaking with Simple Buoy Controller	S>	IMM>	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
11	Logic Level Serial - Serial Handshaking with Simple Buoy Controller	S>	IMM>	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	13	254	5	40	0	0	5	1000	12000	500	5	500	18000	100
12	Logic Level Serial - Hardware Handshaking for Symmetric Communication	S>	IMM>	1	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	13	254	5	40	0	0	5	3000	12000	500	5	500	18000	100
13	Logic Level Serial - Serial Handshaking for Symmetric Communication	S>	IMM>	0	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0	13	254	5	40	0	0	5	1000	12000	500	5	500	18000	100
14	Logic Level Serial - Terminal Program Symmetric Communication	X>	IMM>	0	0	0	1	1	0	1	1	1	0	1	1	0	0	1	0	0	1	0	254	254	5	40	0	0	5	1000	12000	500	5	500	18000	100

13 Electrical Specification

13.1 IMM Connector Description

Connector Option 1. Male vertical for PCB to PCB connection (recommended)
Mating connectors: Through hole - Hirose #DF11-10DS-2DSA(01), DigiKey #H2822-ND
Surface mount - Hirose #DF11-10DS-2V(50), DigiKey #H2837-ND
Connector Option 2. Male Vertical for cable connection
Mating connectors: 10-pin header - Hirose #DF11-10DS-2C, Digikey #H2023-ND
4-pin header - Hirose #DF11-4DS-2C, Digikey #H2019-ND
Crimp contacts (24 to 28 AWG) - Hirose #DF11-2428SCA, DigiKey #H1505-ND
Crimp tool - Hirose #DF11-TA2428HC/US, Digikey #H9924-ND
Pre-crimped wires - DigiKey #H3BBG-10112-B4-ND
Connector Option 3. Male right angle for cable connection
Mating connectors: same as Connector Option 2.

Table 13.1 J1 Pinout RS-232 / Logic Level Serial

Pin #	Signal	Direction	Description
1	IM Coil +	--	Positive side of IM coil (connect to center if using coax)
2	IM Coil -	--	Negative side of IM coil (connect to shield if using coax)
3	HostFlag	In	Hardware Handshake signal. 1.5V to 12V input with internal 4.7K series resistor, 100K pulldown resistor, 10nF load and 12Volt TVS.
4	IMFlag	Out	Hardware Handshake signal. Pulldown only. 100ohm series resistor and 330pF load. Static protected with 12V TVS.
5	LLS_Rx	In	Logic Level Serial receive data input
6	RS-232Rx	In	RS-232 Serial receive data input
7	LLS_Tx	Out	Logic Level Serial transmit data output. Open collector with 100 ohm series resistor and 12Volt TVS
8	RS-232_Tx	Out	RS-232 Serial transmit output.
9	Vin +	--	Positive Power Input
10	Vin - / GND	--	Power Supply Return / Ground

Table 13.2 J1 Pinout RS-485

Pin #	Signal	Direction	Description
1	IM Coil +		Positive side of IM coil (connect to center if using coax)
2	IM Coil -		Negative side of IM coil (connect to shield if using coax)
3	HostFlag	In	Hardware Handshake signal. 1.5V to 12V input with internal 4.7K series resistor, 100K pulldown resistor, 10nF load and 12Volt TVS.
4	IMFlag	Out	Hardware Handshake signal. Pulldown only. 100ohm series resistor and 330pF load. Static protected with 12V TVS.
5	Rx-		RS-485 receive - signal
6	Rx+		RS-485 receive + signal
7	Tx-		RS-485 transmit - signal
8	Tx+		RS-485 transmit + signal
9	Vin +		Power supply 6.0 - 30VDC
10	GND	--	Power Supply Return / Ground

Table 13.3 J2 Pinout (same for RS-232 and RS-485 versions)

Pin #	Signal	Description
1	IM Coil +	Positive side of IM coil (connect to center if using coax)
2	IM Coil -	Negative side of IM coil (connect to shield if using coax)
3	GND	Power supply return (IM coil shield if using twinax cable)
4	Vin +	Power supply 6.0 - 30VDC

13.2 Hardware I/O Line Characteristics

The HostFlag input is triggered on the low-high transition, with one exception. When EnableHostFlagWakeup=1, the IMM tests the state of the HostFlag line just prior to entering Sleep mode. If HostFlag is high, the IMM immediately switches to Host Service instead of going to sleep. If EnableHostFlagWakeup=1, the HostFlag input must be low or floating for the IM to go to sleep.

Table 13.4 Electrical Characteristics

Parameter	Minimum	Typical	Maximum
HostFlag Input, logic low	NA		2.0V
HostFlag Input Pulse width	200uS		
IMFlag Resistance to GND; active		100 ohms	
IMFlag Resistance to GND; inactive	10 megaohms		

13.3 Absolute Maximum Ratings

Table 13.5 Minimum and Maximum Voltages

Parameter	Minimum	Maximum
Power Supply Input Voltage	-30V	30
HostFlag Input Voltage to GND	-10V	+30V
IMFlag pulldown current		30mA
IMFlag Output Voltage to GND, inactive	-0.5V	+12V
Serial Inputs	-12V	+12V
Logic Level Serial input	-12V	+5V

13.4 Power Supply Input

- **Standard Operation** - The IMM power supply is optimized for direct connection to batteries from 9VDC to 24VDC with potential voltage spikes to 30VDC.
- **High Input Voltage Operation** - A High voltage input adapter board is available for applications with input voltages between 20 VDC and 80 VDC. This board holds both an IMM and a high efficiency power converter.

Continuous operation of the IMM at high input voltages without the adapter board is possible at significantly reduced efficiency with the addition of an external series resistor on the power supply input. The series resistor protects the IMM power system from a worst case operating current of approximately 33mA[40]. The series resistor value and wattage rating depend on the maximum input voltage.
Rseries, min = 31 * (Vin,max - 22) ohms (for example, 42V in => 620 ohms)

Vin < 27V => 0.125 Watt resistor rating[41]
Vin < 35V => 0.25 Watt
Vin < 42V => 0.5 Watt
Vin > 42V - not recommended, because the efficiency falls below 40%.

[40] A linear regulator activates in the IMM when the input voltage rises above 19V. The pass element in this regulator should not dissipate more than 150mW. At 35mA the pass element is safe to 24V. For optimum protection, the series resistor must reduce the input voltage below 24V at an input current of 30 mA.
[41] These numbers are for standard axial leaded resistors wrapped in heatshrink, assuming operation in the worst case failure mode for 2 minutes. Continuous operation in the worst case failure mode requires double the series resistor wattage ratings listed.

13.5 Power Consumption Estimates

Power consumption is highly dependent on setup. When connected to a PC with RS-232, the PC's RS-232 load resistors consume more power than the entire IMM. When connected to a low-load RS-232 receiver or with logic-level serial, the device is much more efficient.

Table 13.6 Power Consumption Measurements (Firmware Version 0.98)

Mode	Vin =10V
Asleep (Signal Detector Enabled)[42]	25uA
IM Receive	0.65mA
Host Service (RS232, no load)	0.9 mA
Host Service , RS232 w/PC standard load)	2.15mA
Host Service, Captured line (transmitting) (RS232, no load, low power tx, 100 turn coupler)	1.42mA
IM Service, during reply to # command (RS232, no load, low power tx, 100 turn coupler)	1.42mA

Pessimistic Estimates for an IMM with logic level serial, 12V battery input:

- Total for 1 year, listening to IM line with no activity = 0.3 amp-hours
- Total for 1 year at 1 sample per hour, 2-minute receive per sample = 0.42 amp-hours
- Total for 1 year at 6 samples per hour, 2-minute receive per sample = 1.2 amp-hours

[42] Some early hardware versions draw 33uA in Sleep mode. These devices have serial numbers <70000050.

14 Firmware Revision History

Version 1.09

1. Corrected error in samplegetdata - caused some sample retrieval errors.
2. Added additional data structure protection to IM receive code (should have no visible effect)

Version 1.08

1. Removed testHF code - this test code was not supposed to be released
2. Added <PowerOn/> and <PowerOff/> tags. <PowerOn/> is sent when IMM enters Host Service Mode, <PowerOff/> is sent when IMM exits Host Service Mode.
3. Added SVS monitoring (IMM resets immediately if internal supply voltage falls below 2.5V).

Version 1.07

1. Fixed sample storage error in SampleAddLine, AppendLine, and GData. Sample data was not added correctly.

Version 1.06

1. Fixed TONE SENT in debuglevel>=3 sendwakeuptone (wasn't in an XML comment)
2. Added MLS shortcut for MeasureLineSignal command
3. IMM incorrectly waited for reply to #g00 (was ok for #G00)
4. Added 0.3 second max wait for first @ sign of standard reply - this prevents full TModem2 delay on a quiet line when remote device isn't responding
5. Updated firmware loader
6. Discovery timeout period changed from 5 sec to 8 sec
7. Commands less than 2 characters now generate errors
8. Empty command now generates executed and clears host service mode timer

Version 1.05

1. Added checksums and backup storage of Configuration Data and Hardware data to prevent loss of device configuration if power removed during EEPROM write
2. Corrected error in SampleAdd and SampleAppend commands that caused early termination of data.
3. Corrected timing error in Host Service mode when IM not transmitting. Error resulted in longer fixed delays and longer timeout periods.
4. Removed support for 38400 baud.
5. Added *NOTIMEOUT: command for lab calibration procedures.
6. Added IMMonitor command to log all IM traffic (including noise if no device is transmitting)
7. Added ability to upgrade firmware through RS232 port.
8. Added shortcuts for common commands:
   FCL = ForceCaptureLine
   TCC = TestCableCoupler
   T20CC = Test20TurnCoupler
   REL = ReleaseLine
   SWT = SendWakeupTone

Version 1.04

1. Added filter to transmit voltage measurement
2. Corrected invalid DPSK transmit state after TestCableCoupler command. The invalid state persisted in Host service mode only until the transmitter was disabled, a ! or # command was transmitted or the line was released. The invalid state usually resulted in inappropriate error : FAILED "Low Transmit Voltage - low battery or bad coupler"

Version 1.03

1. All changes of IMFLAG conditional on enableAutoIMFlag setting.
2. Added error for invalid argument on setIMFlag=
3. Warning added to hostfileerase command.
4. Host service 2 min timeout changed to : <TIMEOUT msg='HostService 2 min timeout'/>
5. Event ERROR_IM_CMD_MODE removed.
6. Event ERROR_IMM_BUSY added when the IMM sends a <busy/> signal
7. CRLF from host acts as recognized command, returning <Executed/> and resetting host timeout instead of just giving a new prompt.
8. <Executing/> tags removed from Config Type 1
9. Corrected parsing of !00StayOn command
10. Config Type 1 host serial decoding matched to SIM
11. Time commands removed (hhmmss, mmddyy, ddmmyy)
12. Added !Gx:Mark commands=, sending and receiving
13. Removed ERROR_DPSK_UART_LOCK
14. Added executing tags to captureline reply.
15. Testcablecoupler commands and GETSD no longer cause tx power level to temporarily switch low
16. Interface mode 7 now has enablehostserveonpwrup=0
17. PCBtype >3 enables reset pin. pin must be held low for 1 second on startup

Version 1.02

1. Modified timing of IM replies in Configuration Type 1 ‘bii’ binary commands. This should not cause any performance change.
2. Removed inappropriate TIMEOUT error returned after Configuration Type 1 ‘bii’ binary commands with normal inter-character delay termination.
3. TModem3 maximum value changed to 600 seconds
4. Added TModem4 setting for Config Type 1 use.
5. Config Type 1 bii command inter-character delay timeout now activates after reception of two characters instead of one.
6. Corrected inappropriate logging of Error37 in Config Type 1.
7. Config Type 1 blocked commands now return NOT ALLOWED error.
8. Config Type 1 resets timeout timer on CR or LF, instead of just LF.

Version 1.01

1. Serial Type not preserved when switching from Configuration Type 1 to Type 2.
2. Start of reply marker detection disabled for Configuration Type 1 ‘B’ commands for binary response. This legacy command was used with the UIM, the UIM does not send the start of reply code with the ‘B’ command.

15 Specification Revision History

Version 1.05 2007-03-28

This document supersedes all IMM preliminary specifications.

It describes IMM Firmware Version 1.05. All future changes to the specification will be listed here.

Version 1.09 2008-09-12

Updated firmware revision history

16 Index

A

Adding samples · 31
Appending samples · 31

B

Backspace · 19
Binary data · 30

C

Capturing line · 21
Commands · 36
  detailed descriptions · 40
  group · 37
  sending · 36
  Set · 53
  summary tables · 37
Comment tag · 16
Conceptual model · 6, 8
Configuration Data · 9
Confirmation required tag · 15
Connectors · 58

D

Data · 21
Data reply tag · 16
DPSK Transceiver · 8

E

Echoing · 19
Electrical specification · 58
Erasing samples · 32
Error tag · 14
Event Counters · 9
Executed tag · 14
Executing tag · 16

F

Firmware revision history · 61

G

GData Reply Buffer · 10
Group commands · 37

H

Hardware · 8
Hardware Data · 9
Host · 8
Host Command Buffer · 11
Host Data File · 10
Host Service mode · 12, 18

I

IM Command Buffer · 11
IM Receive mode · 12
IM Reply Buffer · 10
IM Service mode · 12, 25
Inductive Cable Coupler · 8
Interface modes · 33

L

Limited liability statement · 2
Line capturing · 21
Logic level serial modes · 33

M

Mode · 10, 12
Model · 6, 8
Modes
  interface · 33
  logic level serial · 33
  RS-232 · 33

N

Nonvolatile structures · 9

O

Operating mode · 10, 12
Overview · 6
Overwriting samples · 32

P

Pinouts · 58
Power consumption · 60

Power cycling · 12
Power input · 8
Power supply · 59
Prompts · 19

R

Reading samples · 32
Releasing IM line · 21
Remote devices
  receiving commands from · 23
  sending commands to · 21
Reply tags · 14
Resistance · 59
Retrieving list of samples · 31
Revision history
  firmware · 61
  specification · 62
RS-232 modes · 33

S

Sample data · 31
Sample Data · 10
Samples
  adding · 31
  appending · 31
  erasing · 32
  overwriting · 32
  reading · 32
  retrieving list of · 31
Sending commands · 36
Sending data · 21
Signal detector
  disabling · 24
Signal Detector · 8
Sleep mode · 12
Specification
  electrical · 58
Specification revision history · 62
Status Data · 9

T

Tags · 14
THost0 · 28
THost1 · 28
THost2 · 29
THost3 · 29
THost4 · 29
THost5 · 29
Timeout tag · 16
Timeouts · 19, 28
TModem2 · 20
TModem3 · 20
TModem4 · 20

V

Volatile structures · 10
Voltage · 59

W

Wake-up tone · 21, 23
Waking IMM · 23
Warning tag · 15

X

XML-like format · 14